iNTERFACEWARE Inc.
672 Dupont Street, Suite 505
Toronto, ON, M6G 1Z6
interfaceware.com

Juggling Internal & External APIs with Iguana Exercise

For IUC2019

September 19th, 2019

Purpose

This document describes the exercise accompanying the *Juggling Internal & External APIs with Iguana* presentation at IUC2019.

Included is a GitHub repository that will contain two channels, *Exercise2: API Client* and *Exercise2: API Server*, that can be imported into any Iguana instance (see Add/Configure Repositories and Import Channels).

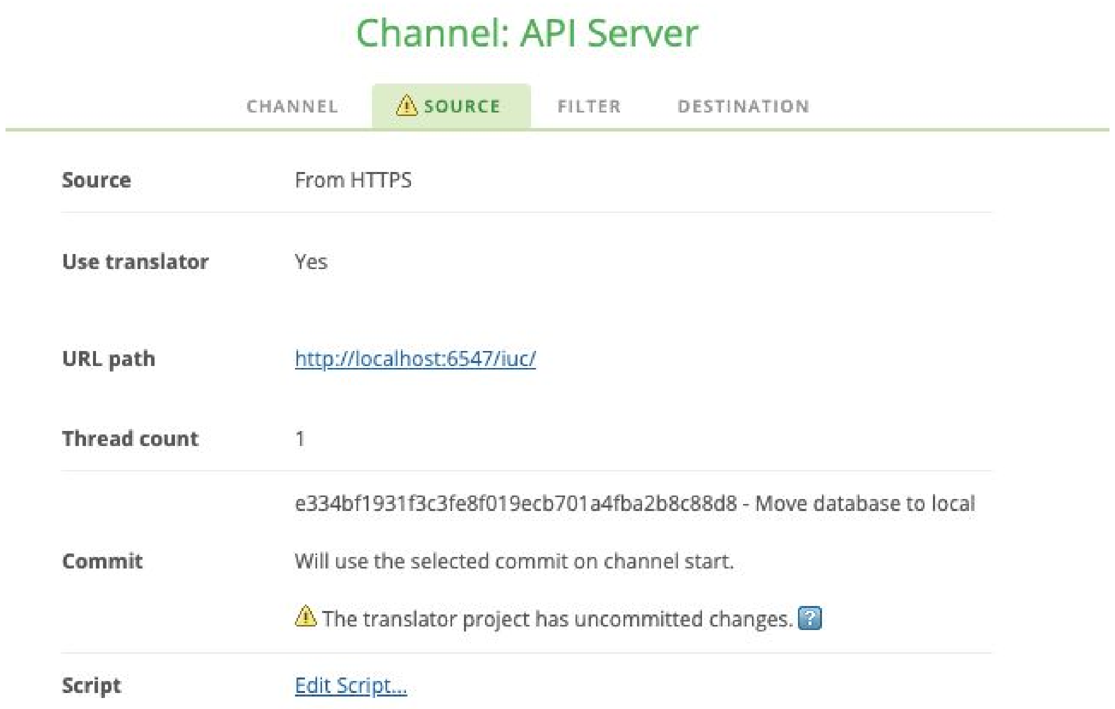
If you have any questions or concerns, please contact us at support@interfaceware.com and CC paul.le@interfaceware.com in the email.

API Server

The *Exercise2: API Server* channel provides an example of an API server generated from an API created in the API Designer. For more detailed information on how to use the API Designer, please refer to the documentation for the API Designer.

Generating an API Token

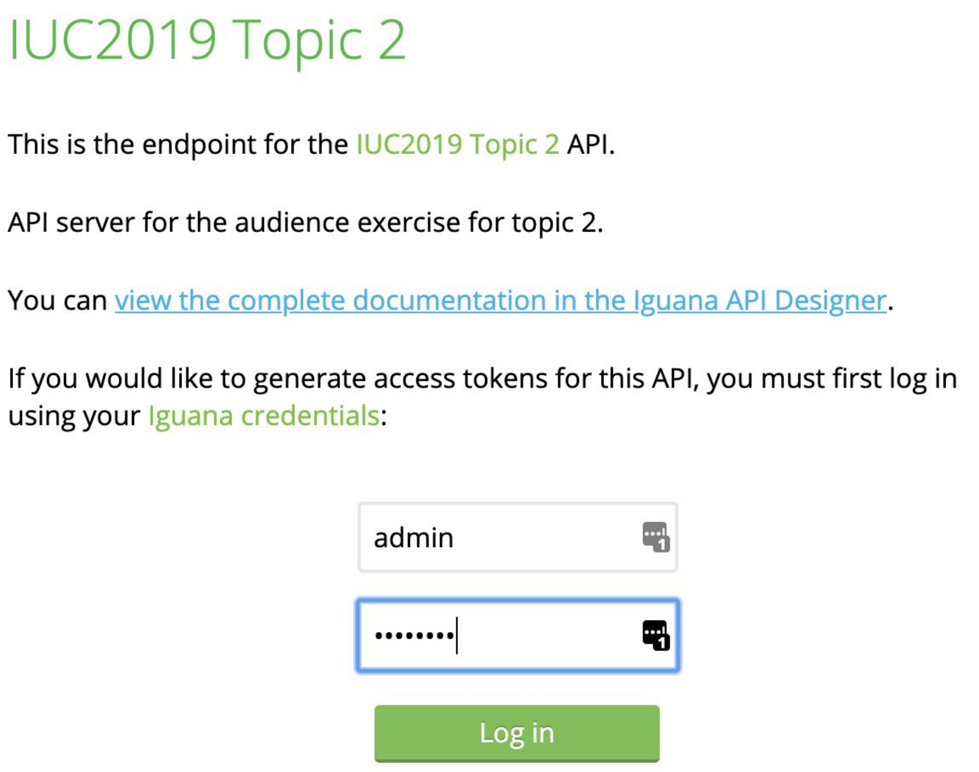
1. Once the *Exercise2: API Server* channel has been imported, start the channel and navigate to the URL path for that channel:



Channel: API Server

Source	From HTTPS
Use translator	Yes
URL path	http://localhost:6547/iuc/
Thread count	1
Commit	e334bf1931f3c3fe8f019ecb701a4fba2b8c88d8 - Move database to local Will use the selected commit on channel start. ⚠ The translator project has uncommitted changes.
Script	Edit Script...

2. Enter in the credentials for an Iguana user:



IUC2019 Topic 2

This is the endpoint for the IUC2019 Topic 2 API.

API server for the audience exercise for topic 2.

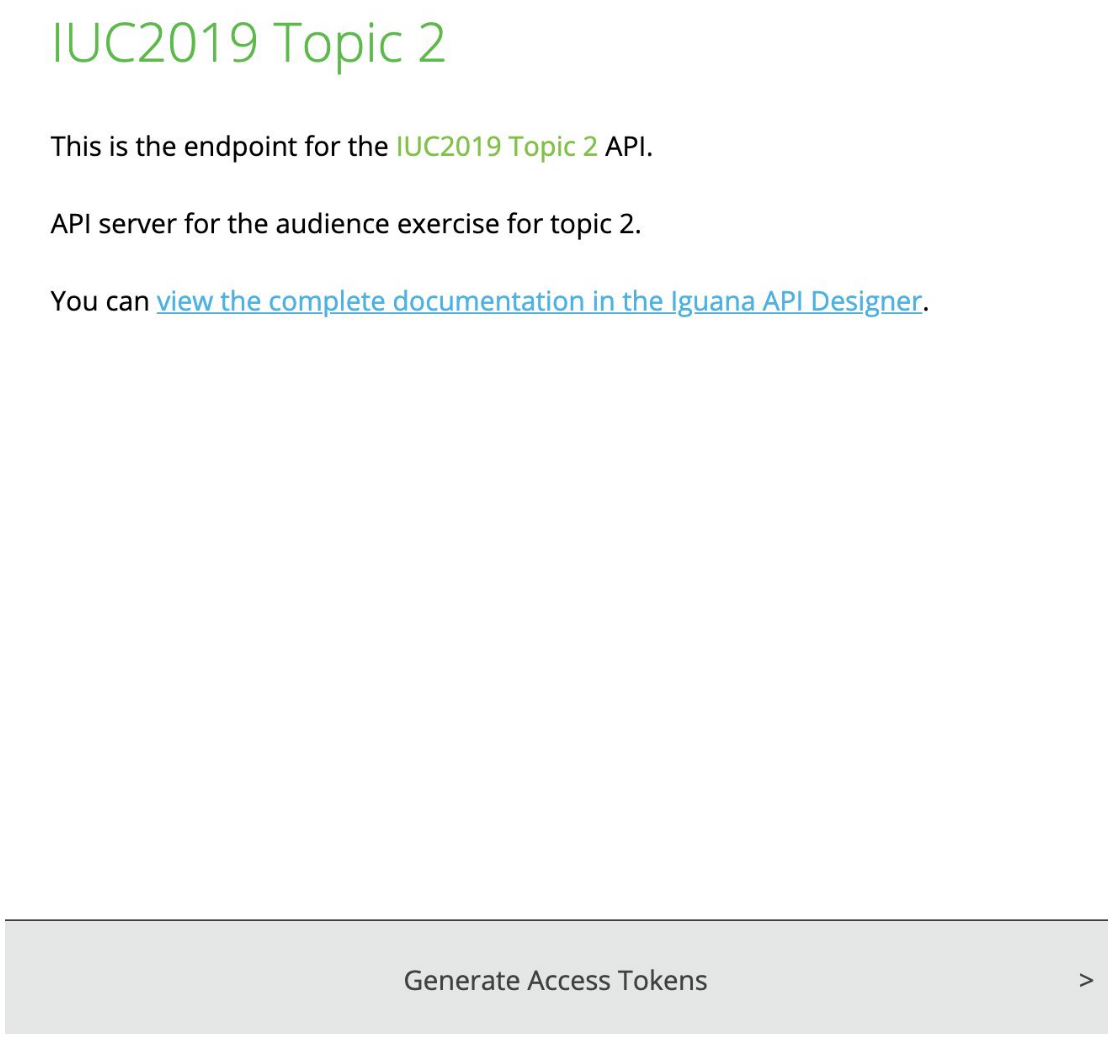
You can view the complete documentation in the Iguana API Designer.

If you would like to generate access tokens for this API, you must first log in using your Iguana credentials:

admin

••••••••

3. Click on *Generate Access Token*:



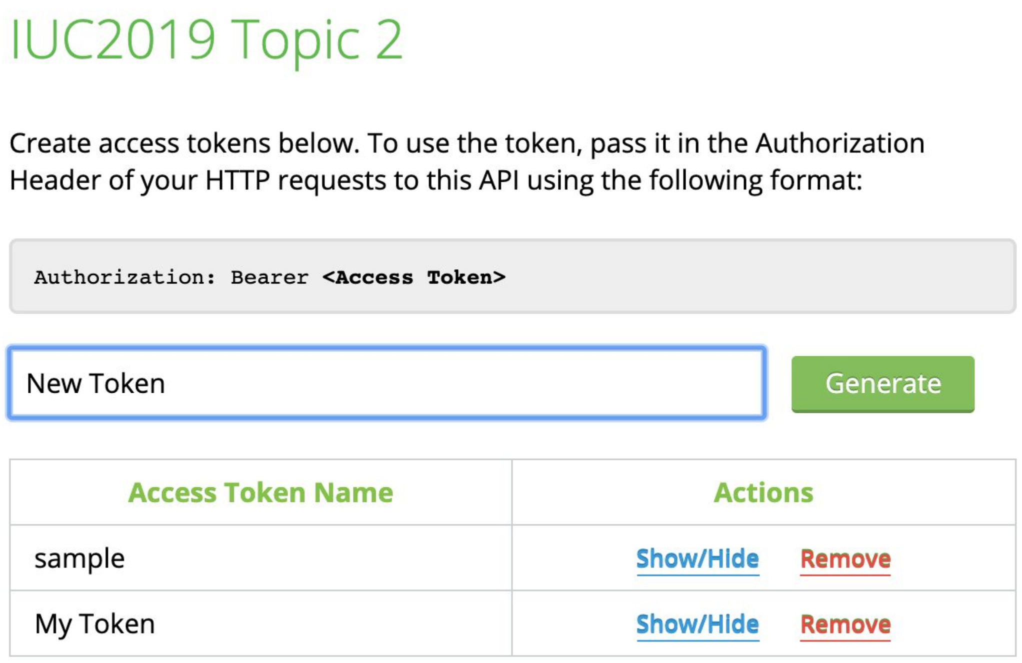
IUC2019 Topic 2

This is the endpoint for the IUC2019 Topic 2 API.

API server for the audience exercise for topic 2.

You can view the complete documentation in the Iguana API Designer.

4. Enter a unique name for the access token, and click *Generate*:



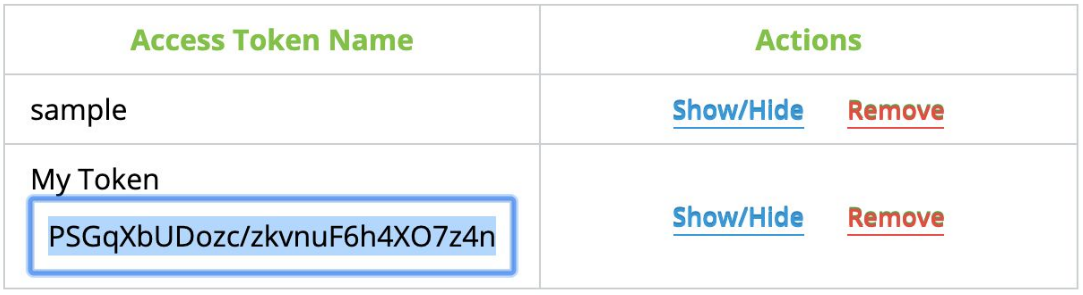
IUC2019 Topic 2

Create access tokens below. To use the token, pass it in the Authorization Header of your HTTP requests to this API using the following format:

```
Authorization: Bearer <Access Token>
```

Access Token Name	Actions
sample	Show/Hide Remove
My Token	Show/Hide Remove

5. These access token can be used to make authenticated requests to the *Exercise2: API Server* channel:



Access Token Name	Actions
sample	Show/Hide Remove
My Token PSGqXbUDozc/zkvnuF6h4XO7z4n	Show/Hide Remove

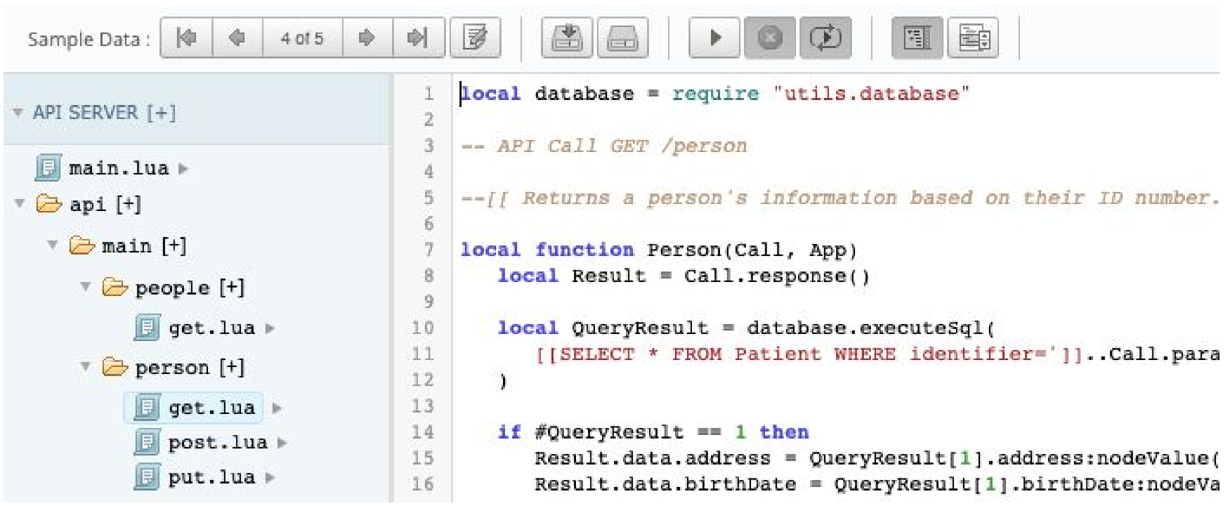
Looking at the API

Included is a JSON file called *IUC2019 API Server.grammar.json*. In order to view this API grammar on the API Designer, please follow the steps below:

1. Log into the API designer
2. Go to My APIs
3. Click the + *Import API* link at the bottom of the left pane, and import the grammar file
4. Name the imported API as "*IUC2019 API Server*"
5. View the Resources
6. View the Interactions

Looking at the Code

Once the *Exercise2: API Client* channel has been imported from the included repository, navigate to the Translator script of the *From HTTPS* component. The *api/main* directory will contain Lua modules that correspond to the API endpoints defined in the API Designer for *IUC2019 API Server*:



```
local database = require "utils.database"

-- API Call GET /person

--[[ Returns a person's information based on their ID number.

local function Person(Call, App)
   local Result = Call.response()

   local QueryResult = database.executeSql(
      [[SELECT * FROM Patient WHERE identifier=']]..Call.para
   )

   if #QueryResult == 1 then
      Result.data.address = QueryResult[1].address:nodeValue(
      Result.data.birthDate = QueryResult[1].birthDate:nodeVa
```

API Client

The *Exercise2: API Client* channel walks through the following examples of how to handle API authentication with Iguana:

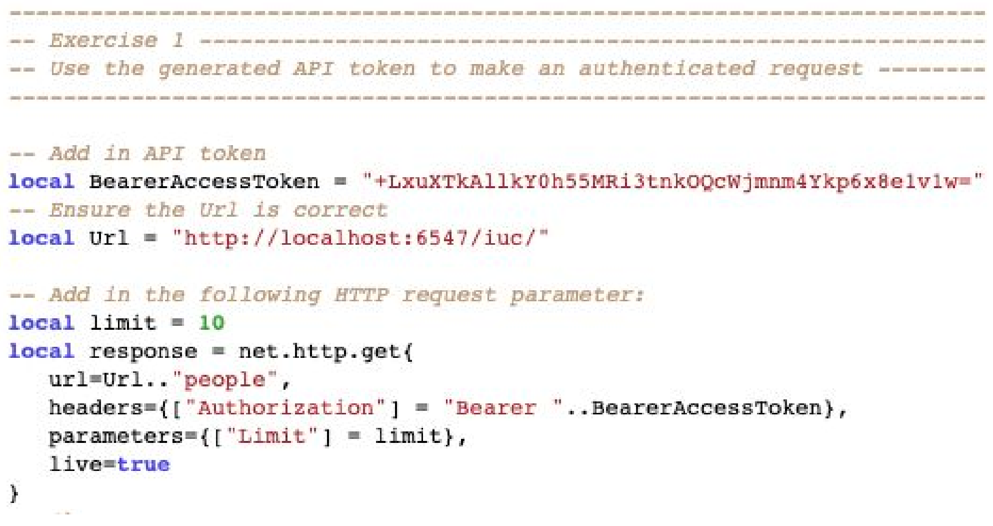
1. HTTP Basic Authentication

2. API Keys

Looking at the Code

Once the *Exercise2: API Client* channel has been imported, navigate to the Translator script of the *From Translator* component. The *main.lua* module will contain Lua script demonstrating examples of the API authentication methods mentioned above.

Ensure that the *Exercise2: API Server* channel has been imported as well from [the GitHub repository](), and that there is an access key available (see *Generating an API Token* above). In the channel code, add the access key to the *BearerAccessToken* variable and ensure the URL is pointing to the correct *Exercise2: API Server* channel URL:



```
--------------------------------------------------------------------------
-- Exercise 1 ------------------------------------------------------------
-- Use the generated API token to make an authenticated request --------
--------------------------------------------------------------------------

-- Add in API token
local BearerAccessToken = "+LxuXTkAllkY0h55MRi3tnkOQcWjmnm4Ykp6x8e1v1w="
-- Ensure the Url is correct
local Url = "http://localhost:6547/iuc/"

-- Add in the following HTTP request parameter:
local limit = 10
local response = net.http.get{
   url=Url.."people",
   headers={["Authorization"] = "Bearer "..BearerAccessToken},
   parameters={["Limit"] = limit},
   live=true
}
```