The Virtual Forests API

Guillermo Vega Gorgojo

June 2022

1. Introduction

This document is a brief guide for accessing the API of the RDF version of the REINFORCE dataset (as part of the work in the Virtual Forests project).

The REINFORCE dataset is available at this public endpoint: https://crossforest.gsic.uva.es/pruebas/sparql/.

The API aims to facilitate the access to the REINFORCE dataset. In particular, prospective users should not need to know Semantic Web technologies such as RDF and SPARQL. Nevertheless, some basic knowledge of the following topics is recommended (check the resources):

- REST APIs
  - Learn REST: A RESTful Tutorial, by Todd Fredrich
  - REST: a FAQ, by Diogo Lucas
  - RFC 2616
- JSON
  - RFC 8259
- URIs and IRIs
  - RFC 3986
  - RFC 3987

In order to use the proposed API you can use Postman, a GUI REST client. This client allows you to save both calls and responses, is free, is easy to configure, and works on both Mac and PC, as well as in modern web browsers such as Firefox or Chrome.

2. The Virtual Forests API in a nutshell

The Virtual Forests API is accessible at https://crafts.gsic.uva.es/apis/virfor

This API exposes the following operations:

- **GET** `https://crafts.gsic.uva.es/apis/virfor/`
  - Obtain the configuration file of this API
- **GET** `https://crafts.gsic.uva.es/apis/virfor/resource?id={id}&iri={iri}`
  - Obtain the representation of a resource of type `id` (such as `Arboretum` or `SurvivalRate`, as described in the configuration file) and IRI `iri`
- **GET** `https://crafts.gsic.uva.es/apis/virfor/resources?id={id}&iris={iriA}&iris={iriB}...`
  - Obtain the representation of multiple resources with IRIS `iriA`, `iriB` ... of type `id`
- **GET** `https://crafts.gsic.uva.es/apis/virfor/query?id={id}&{parA}={valA}&{parB}={valB}...`
  - Send a parametrized query with id `id` and parameters, e.g. `parA`, and corresponding values, e.g. `valA`, as required by the query (check the configuration file)

These operations require so-called *Bearer authentication* with token `4825bb0d-c535-4d2c-9237-2bd1daf452ab`.

3. First steps with Postman and the API

1. Go to https://www.postman.com/downloads/ and get the Postman app or try the Web version (this is the one I will use)

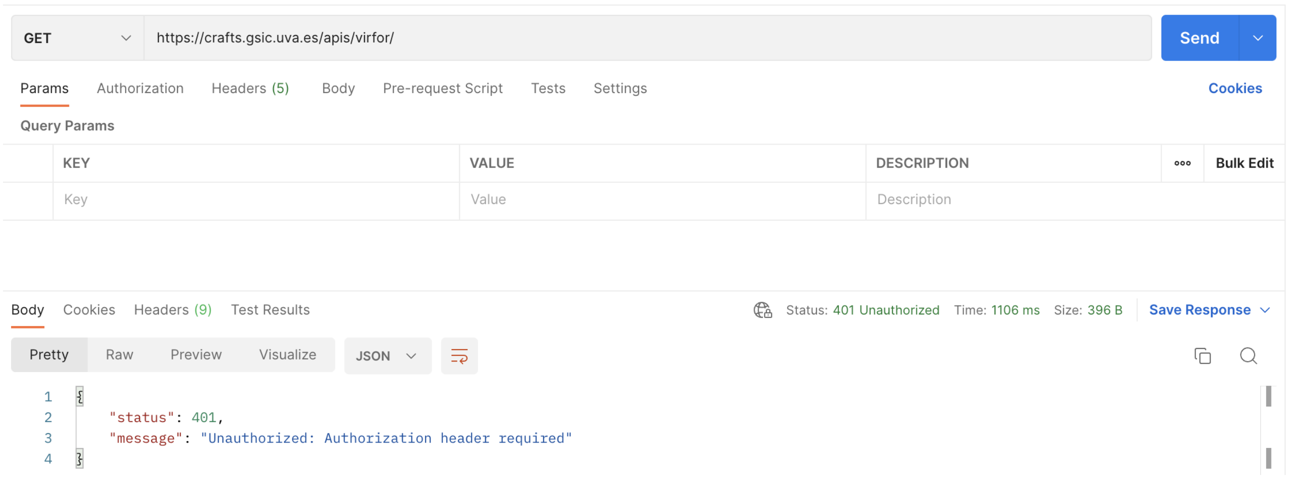
2. Create an account and sign in when prompted

3. If this is your first time launching Postman, a welcome screen appears. Click **Create new** in order to create a new request

4. Insert the following URI into the box next to **GET**: `https://crafts.gsic.uva.es/apis/virfor/`

5. Click **Send**

   The response appears in the lower pane. For example:



https://crafts.gsic.uva.es/apis/virfor/

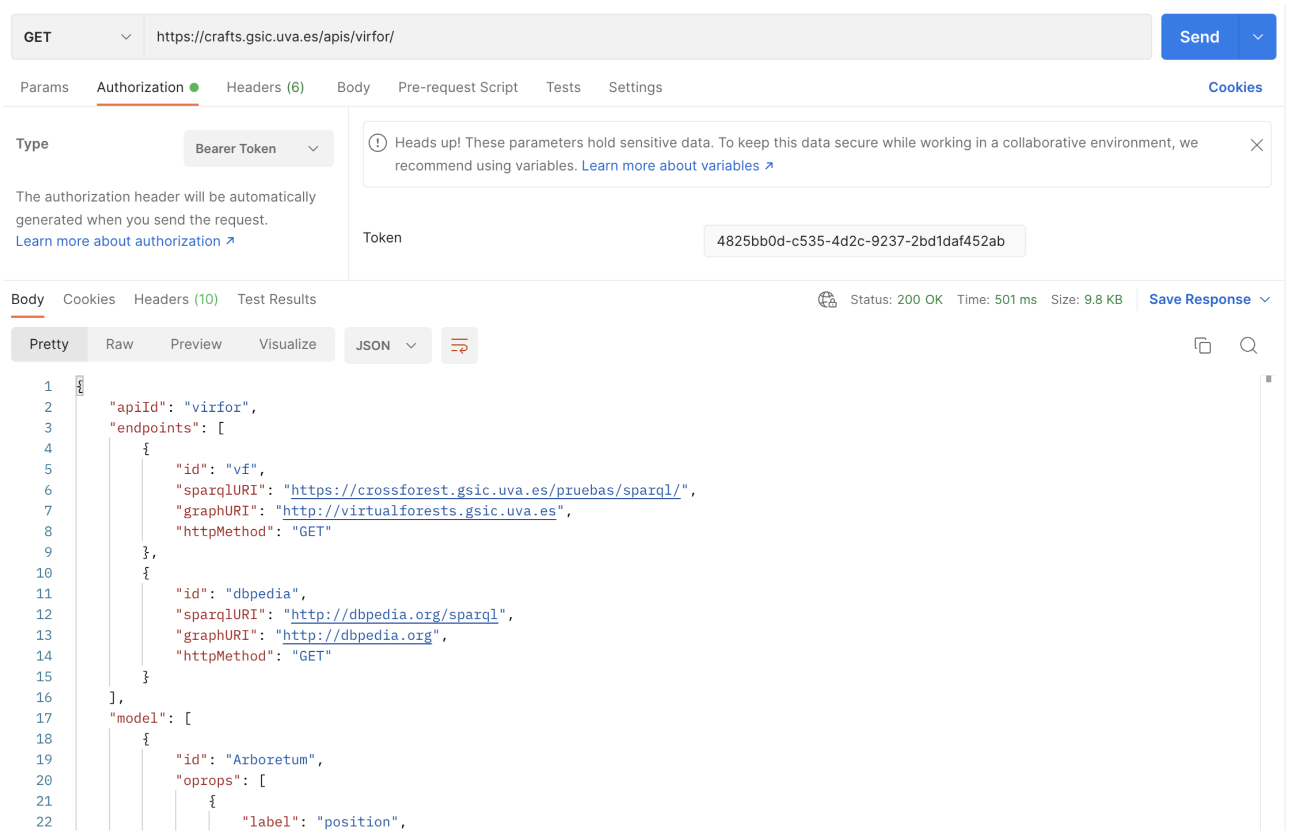
Status: 401 Unauthorized Time: 1106 ms Size: 396 B

```
{
    "status": 401,
    "message": "Unauthorized: Authorization header required"
}
```

6. You are unauthorized to perform such operation!

   Click the **Authorization** tab, select **Bearer Token** as type, and then use `4825bb0d-c535-4d2c-9237-2bd1daf452ab` as **Token**.

   Click **Send** and you will get a valid response:



https://crafts.gsic.uva.es/apis/virfor/

Type: Bearer Token

The authorization header will be automatically generated when you send the request.
Learn more about authorization ↗

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables ↗

Token: 4825bb0d-c535-4d2c-9237-2bd1daf452ab

Status: 200 OK Time: 501 ms Size: 9.8 KB

```
{
    "apiId": "virfor",
    "endpoints": [
        {
            "id": "vf",
            "sparqlURI": "https://crossforest.gsic.uva.es/pruebas/sparql/",
            "graphURI": "http://virtualforests.gsic.uva.es",
            "httpMethod": "GET"
        },
        {
            "id": "dbpedia",
            "sparqlURI": "http://dbpedia.org/sparql",
            "graphURI": "http://dbpedia.org",
            "httpMethod": "GET"
        }
    ],
    "model": [
        {
            "id": "Arboretum",
            "oprops": [
                {
                    "label": "position",
```

The response is the configuration file that I've prepared to set up the API.

4. A quick look to the configuration file

The configuration file is a long JSON object with the instructions to access the source datasets through a REST API. Don't worry, I'll just cover the basics to understand what is about and how you can use it to access the data.

The configuration file has the following keys:

- `apiId` : the name of the API (value `virfor`)
- `endpoints` : an array of the data sources with their access information. The primary source is `vf` , containing the REINFORCE dataset. `dbpedia` is a secondary source that is only employed to gather additional information about species
- `model` : an array with all the resource types exposed by the API. Each resource type includes:
    - An `id` such as `Arboretum` , `Position` , `Provenance` , etc.
    - Several attributes within the arrays `oprops` , `dprops` , and `types` . Here you should only care about the `label` (corresponding to the attribute name); the rest of information is employed for extracting the data from the sources
- `queryTemplates` : an array with a number of query templates that use an expert query language, SPARQL. Happily, you don't need to know SPARQL in order to use these templates (you'll see later with examples). Each template includes:
    - An `id` such as `taxa` , `provenances` , `arboreta` , etc.
    - A textual `description` of the template (read it to grasp what is the template purposed for)
    - The actual `template` . This is for the query engine, so you don't need to read it
    - The response of a query is essentially a table, the column names correspond to the `variables`
    - A template can be parametrized providing values to the declared `parameters` . Note that parameters can be optional and have an expected type

5. Using the API to retrieve representations of resources

It is very easy to get a representation of a resource with a known IRI. This just requires a GET operation with this format: `https://crafts.gsic.uva.es/apis/virfor/resource?id={id}&iri={iri}`

You only have to replace `{id}` with the resource type and `{iri}` with the IRI of the resource. Let's try with an example, we have an arboretum with IRI `http://reinfforce.iefc.net/data/arboretum/AR01` :

1. Check the configuration file `https://crafts.gsic.uva.es/apis/virfor/` in Postman to identify the id of the resource type: `Arboretum`

2. Craft the URI of the GET operation following the format above:

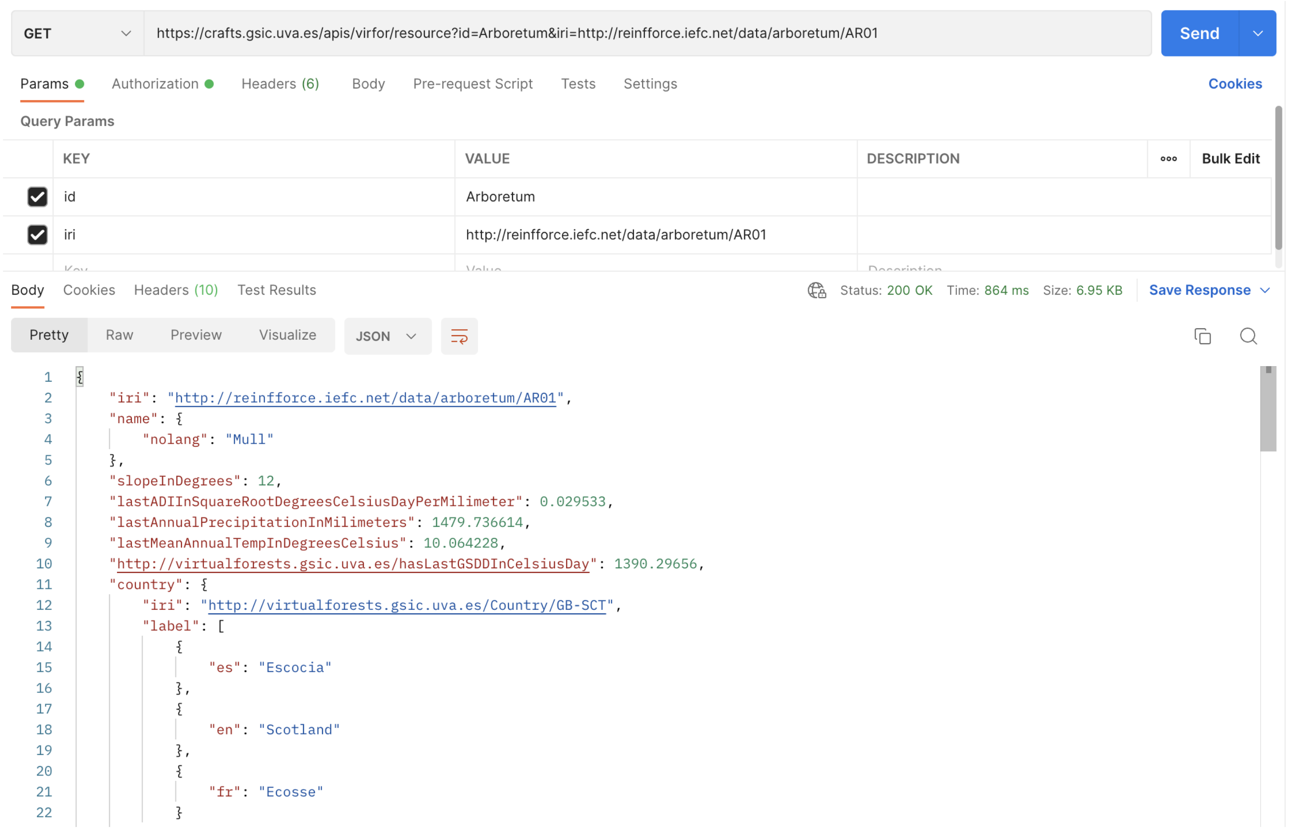
   `https://crafts.gsic.uva.es/apis/virfor/resource?id=Arboretum&iri=http://reinfforce.iefc.net/data/arboretum/AR01`

3. Click **Create new** in Postman to create a new request

4. Insert the crafted URI into the box next to **GET**

5. Click the **Authorization** tab, select **Bearer Token** as type, and then use `4825bb0d-c535-4d2c-9237-2bd1daf452ab` as **Token**

6. Click **Send** to get your response:



GET https://crafts.gsic.uva.es/apis/virfor/resource?id=Arboretum&iri=http://reinfforce.iefc.net/data/arboretum/AR01

Query Params

KEY	VALUE	DESCRIPTION
id	Arboretum	
iri	http://reinfforce.iefc.net/data/arboretum/AR01	

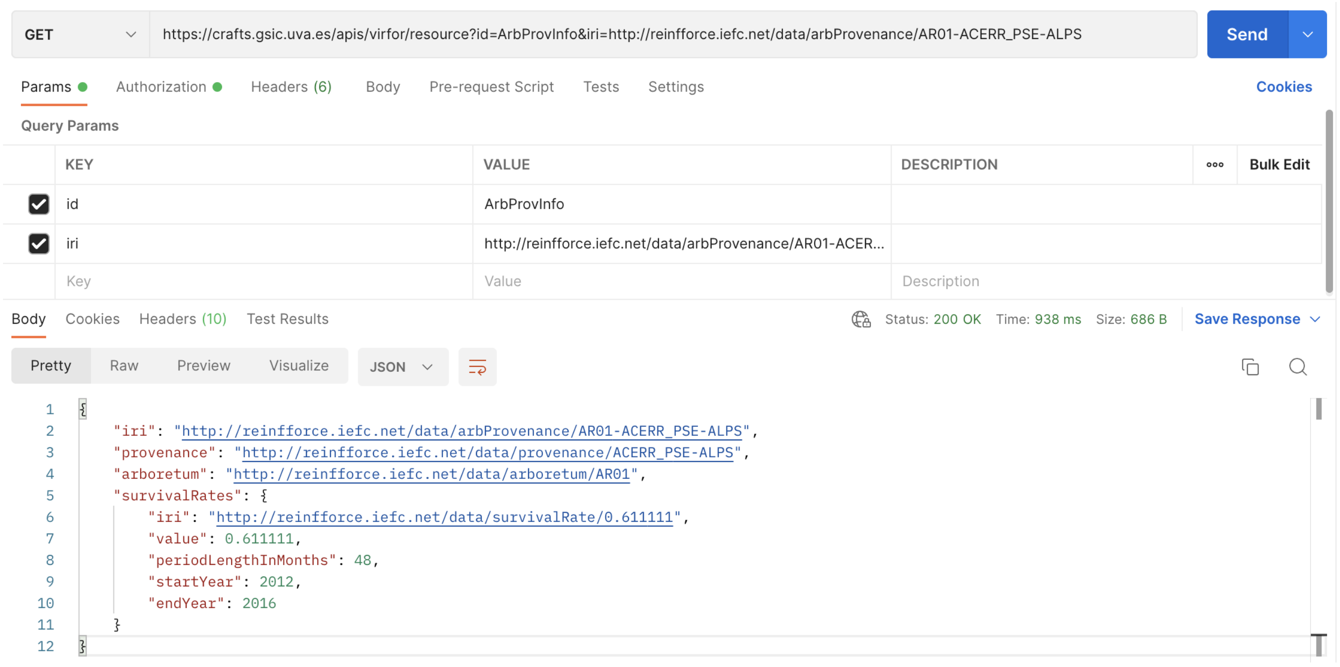
Status: 200 OK Time: 864 ms Size: 6.95 KB

```
{
    "iri": "http://reinfforce.iefc.net/data/arboretum/AR01",
    "name": {
        "nolang": "Mull"
    },
    "slopeInDegrees": 12,
    "lastADIInSquareRootDegreesCelsiusDayPerMilimeter": 0.029533,
    "lastAnnualPrecipitationInMilimeters": 1479.736614,
    "lastMeanAnnualTempInDegreesCelsius": 10.064228,
    "http://virtualforests.gsic.uva.es/hasLastGSDDInCelsiusDay": 1390.29656,
    "country": {
        "iri": "http://virtualforests.gsic.uva.es/Country/GB-SCT",
        "label": [
            {
                "es": "Escocia"
            },
            {
                "en": "Scotland"
            },
            {
                "fr": "Ecosse"
            }
```

The response is a JSON object with the same keys defined for an `Arboretum` in the model of the API.

If you check the response, you will find a lot of information about that arboretum. Specifically, there is a list of provenance info objects such as `http://reinfforce.iefc.net/data/arbProvenance/AR01-ACERR_PSE-ALPS`. You can use this IRI to make a new request and gather more information (you need to set the right `id` parameter by checking the API model, it is `ArbProvInfo` in this case):

`https://crafts.gsic.uva.es/apis/virfor/resource?id=ArbProvInfo&iri=http://reinfforce.iefc.net/data/arbProvenance/AR01-ACERR_PSE-ALPS`



GET https://crafts.gsic.uva.es/apis/virfor/resource?id=ArbProvInfo&iri=http://reinfforce.iefc.net/data/arbProvenance/AR01-ACERR_PSE-ALPS

Query Params

KEY	VALUE	DESCRIPTION
id	ArbProvInfo	
iri	http://reinfforce.iefc.net/data/arbProvenance/AR01-ACER...	
Key	Value	Description

Status: 200 OK Time: 938 ms Size: 686 B

```
{
    "iri": "http://reinfforce.iefc.net/data/arbProvenance/AR01-ACERR_PSE-ALPS",
    "provenance": "http://reinfforce.iefc.net/data/provenance/ACERR_PSE-ALPS",
    "arboretum": "http://reinfforce.iefc.net/data/arboretum/AR01",
    "survivalRates": {
        "iri": "http://reinfforce.iefc.net/data/survivalRate/0.611111",
        "value": 0.611111,
        "periodLengthInMonths": 48,
        "startYear": 2012,
        "endYear": 2016
    }
}
```

Here you can see the survival rate of provenance `http://reinfforce.iefc.net/data/provenance/ACERR_PSE-ALPS` for arboretum `http://reinfforce.iefc.net/data/arboretum/AR01`. If you are wondering which provenance is this, we just need to made a new request with `id` equal to `Provenance`:

```
https://crafts.gsic.uva.es/apis/virfor/resource?id=Provenance&iri=http://reinfforce.iefc.net/data/provenance/ACERR_PSE-ALPS
```

You can check that this is *Acer pseudoplatanus* from Alps Switzerland. There is a lot of additional information about the species, e.g. an image and Wikipedia descriptions in multiple languages, as well as a list of arboretum-provenance objects.

6. Retrieve multiple representations of resources with one call

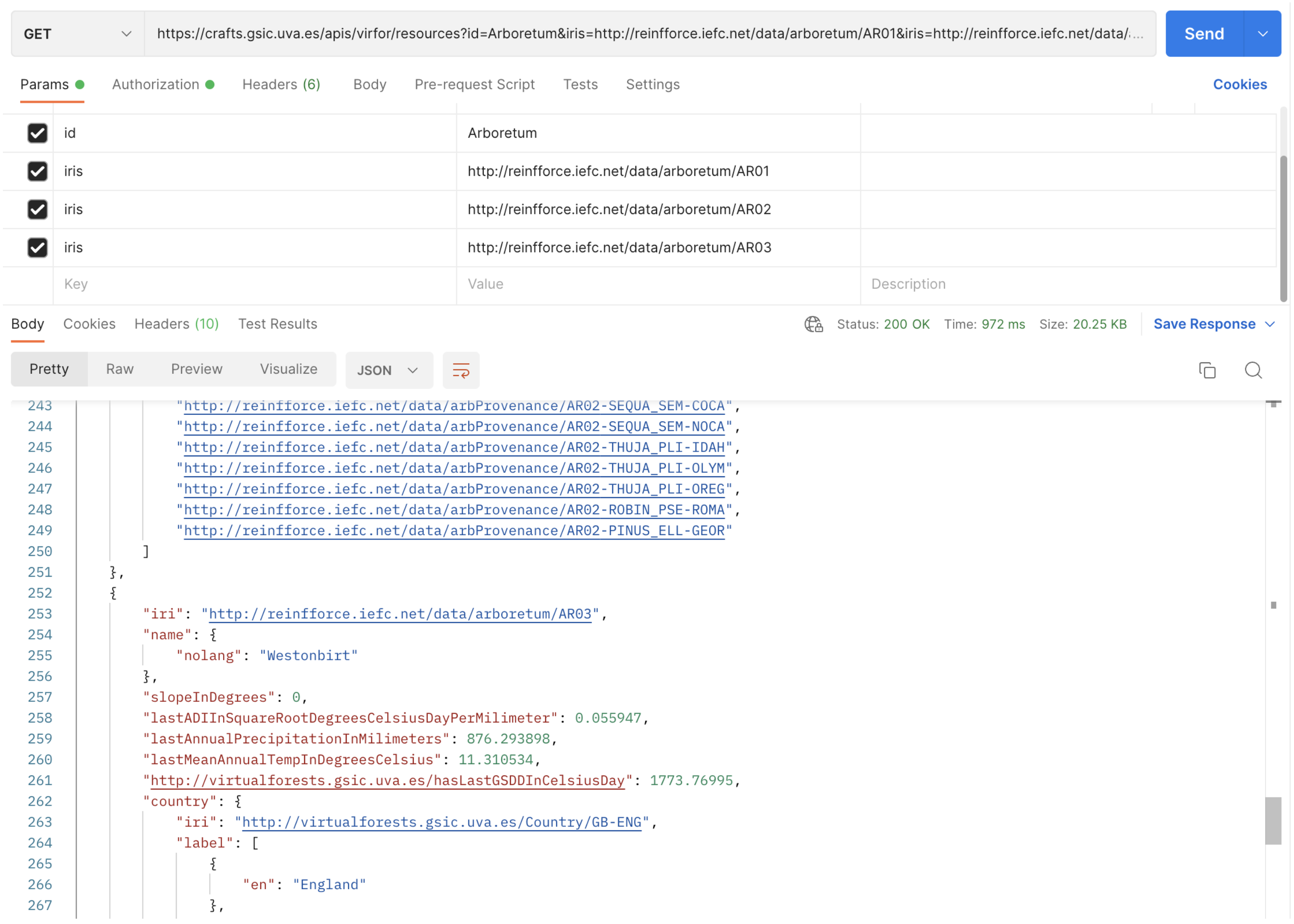
Imagine that you have many resources of the same type, e.g. arboreta `http://reinfforce.iefc.net/data/arboretum/AR01` , `http://reinfforce.iefc.net/data/arboretum/AR02` , and `http://reinfforce.iefc.net/data/arboretum/AR03` .

You can retrieve all their representations with just one call. The format of the corresponding GET operation is:

```
https://crafts.gsic.uva.es/apis/virfor/resources?id={id}&iris={iriA}&iris={iriB}...
```

In our example:

```
https://crafts.gsic.uva.es/apis/virfor/resources?id=Arboretum&iris=http://reinfforce.iefc.net/data/arboretum/AR01&iris=http://reinfforce.iefc.net/data/arboretum/AR02&iris=http://reinfforce.iefc.net/data/arboretum/AR03
```



https://crafts.gsic.uva.es/apis/virfor/resources?id=Arboretum&iris=http://reinfforce.iefc.net/data/arboretum/AR01&iris=http://reinfforce.iefc.net/data/...

	Key	Value	Description
✓	id	Arboretum	
✓	iris	http://reinfforce.iefc.net/data/arboretum/AR01	
✓	iris	http://reinfforce.iefc.net/data/arboretum/AR02	
✓	iris	http://reinfforce.iefc.net/data/arboretum/AR03	

Status: 200 OK Time: 972 ms Size: 20.25 KB

```
            "http://reinfforce.iefc.net/data/arbProvenance/AR02-SEQUA_SEM-COCA",
            "http://reinfforce.iefc.net/data/arbProvenance/AR02-SEQUA_SEM-NOCA",
            "http://reinfforce.iefc.net/data/arbProvenance/AR02-THUJA_PLI-IDAH",
            "http://reinfforce.iefc.net/data/arbProvenance/AR02-THUJA_PLI-OLYM",
            "http://reinfforce.iefc.net/data/arbProvenance/AR02-THUJA_PLI-OREG",
            "http://reinfforce.iefc.net/data/arbProvenance/AR02-ROBIN_PSE-ROMA",
            "http://reinfforce.iefc.net/data/arbProvenance/AR02-PINUS_ELL-GEOR"
        ]
    },
    {
        "iri": "http://reinfforce.iefc.net/data/arboretum/AR03",
        "name": {
            "nolang": "Westonbirt"
        },
        "slopeInDegrees": 0,
        "lastADIInSquareRootDegreesCelsiusDayPerMilimeter": 0.055947,
        "lastAnnualPrecipitationInMilimeters": 876.293898,
        "lastMeanAnnualTempInDegreesCelsius": 11.310534,
        "http://virtualforests.gsic.uva.es/hasLastGSDDInCelsiusDay": 1773.76995,
        "country": {
            "iri": "http://virtualforests.gsic.uva.es/Country/GB-ENG",
            "label": [
                {
                    "en": "England"
                },
```

7. How can I know the IRIs of resources?

We have seen how we can use the API to retrieve representations of resources. However, this requires knowing somehow their IRIs in advance. Query templates are intended to fulfil this need. As you can see in the config file, I have prepared a number of query templates. Let's examine how to work with them.

First of all, this is the format of the GET operation for query templates:

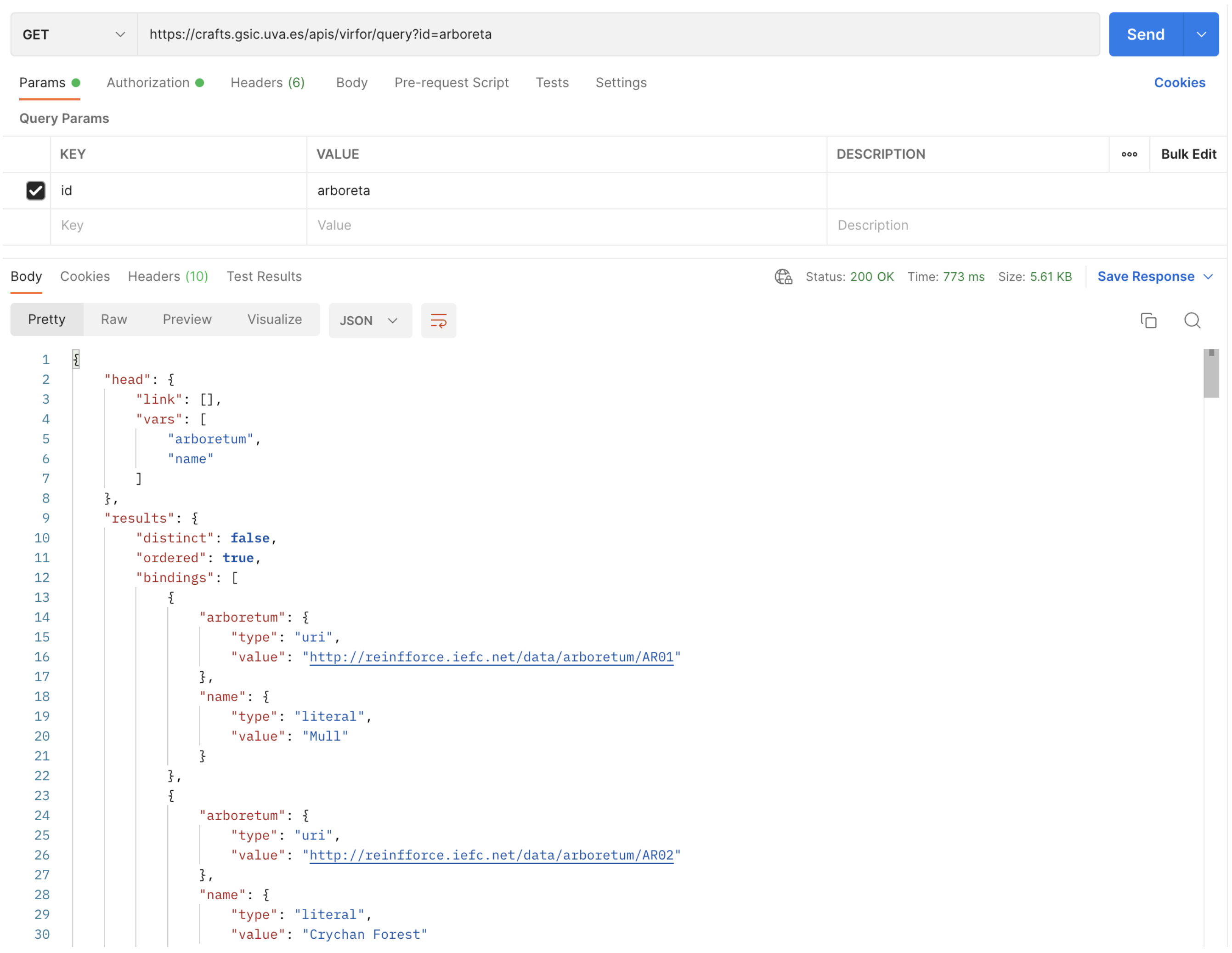
```
https://crafts.gsic.uva.es/apis/virfor/query?id={id}&{parA}={valA}&{parB}={valB}...
```

Note that the `id` can be found in the config file, while suitable parameters and values are set for the query template at hand. Let's try with the query template `arboreta` (check the config file):

```
{
        "id": "arboreta",
        "description": "Obtain all the arboreta (variable \"arboretum\") with their local names
            (variable \"name\") in the Virtual Forests repository",
        "template": "select distinct ?arboretum ?name where {\n  ?arboretum a <http://virtualforests.gs
        "variables": [
          "arboretum",
          "name"
        ],
        "parameters": [
        ],
        "endpoint": "vf"
}
```

This is a simple query template with no parameters. We can use it to get the list of the arboreta in the dataset:

```
https://crafts.gsic.uva.es/apis/virfor/query?id=arboreta
```


GET https://crafts.gsic.uva.es/apis/virfor/query?id=arboreta

Query Params

KEY	VALUE	DESCRIPTION
id	arboreta	

Status: 200 OK Time: 773 ms Size: 5.61 KB

```
{
    "head": {
        "link": [],
        "vars": [
            "arboretum",
            "name"
        ]
    },
    "results": {
        "distinct": false,
        "ordered": true,
        "bindings": [
            {
                "arboretum": {
                    "type": "uri",
                    "value": "http://reinfforce.iefc.net/data/arboretum/AR01"
                },
                "name": {
                    "type": "literal",
                    "value": "Mull"
                }
            },
            {
                "arboretum": {
                    "type": "uri",
                    "value": "http://reinfforce.iefc.net/data/arboretum/AR02"
                },
                "name": {
                    "type": "literal",
                    "value": "Crychan Forest"
```

We get the answer directly from the SPARQL endpoint in JSON-LD format. It is a regular JSON object, although a bit more verbose than needed. Anyway, the important thing to look is the `bindings` array. Each object in the array is a valid answer (a row of the table of results) with the same fields defined in the query template. You can browse the list of species and get the IRIs of the ones you are interested, e.g. `http://reinfforce.iefc.net/data/arboretum/AR01` is the IRI of arboretum **Mull**.

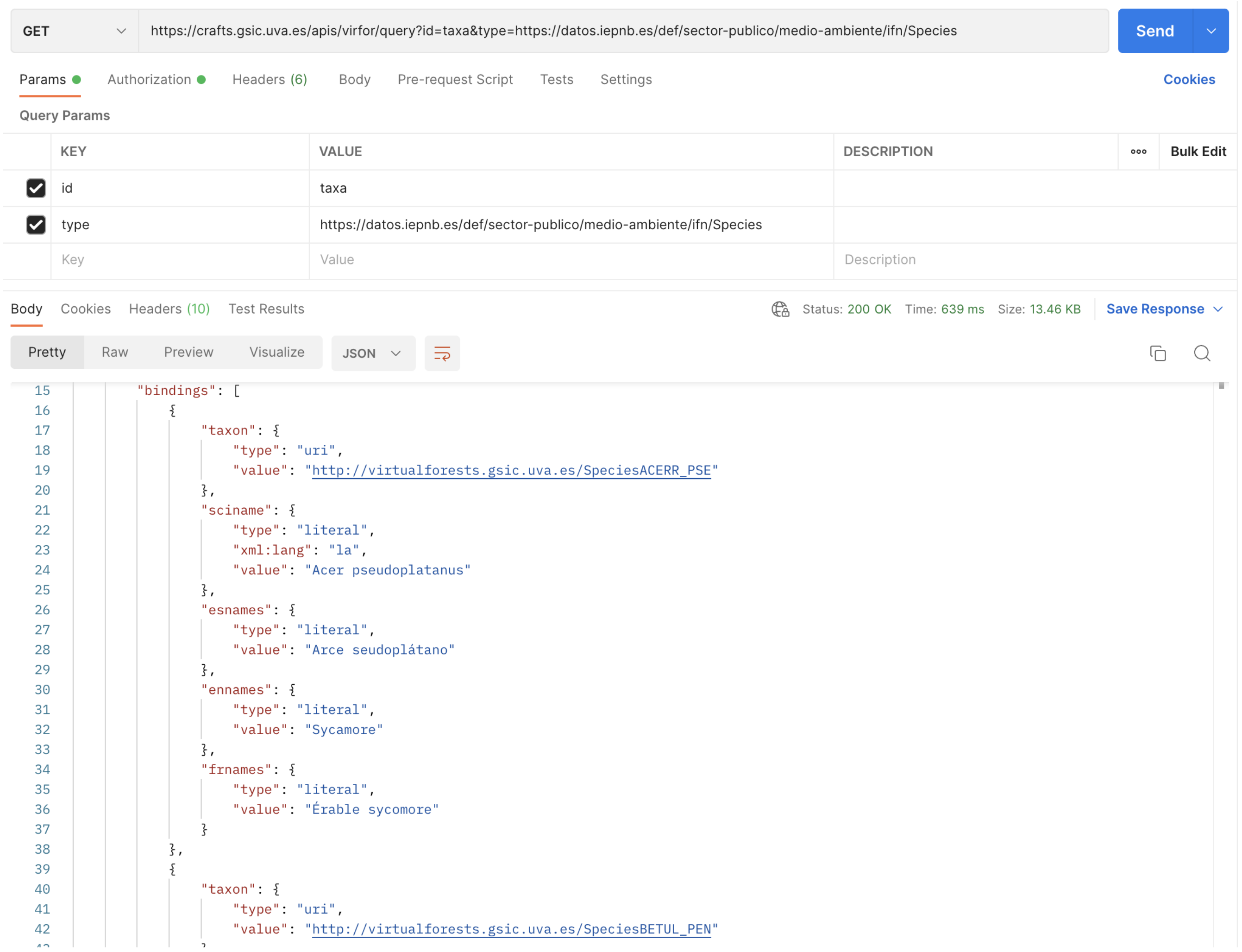
We can move on to the query template `taxa` (check the config file):

```
{
  "id": "taxa",
  "description": "Obtain all the taxa (variable \"taxon\") with their scientific names
     (variable \"sciname\"), their Spanish vulgar names (variable \"esnames\"), their
     French vulgar names (variable \"frnames\"), and their English vulgar names
     (variable \"ennames\") in the Virtual Forests repository. Taxa can be filtered by
     an optional type (variable \"type\") corresponding to https://datos.iepnb.es/def/sector-publico/
  "template": "select distinct ?taxon ?sciname (group_concat(distinct ?esname;separator=\"; \") as ?e
  "variables": [
     "taxon",
     "sciname",
     "esnames",
     "ennames",
     "frnames"
  ],
  "parameters": [
    {
       "label": "type",
       "type": "iri",
       "optional": true
    }
  ],
  "endpoint": "vf"
}
```

This query template has an optional parameter `type` for filtering taxa by Family `https://datos.iepnb.es/def/sector-publico/medio-ambiente/ifn/Family` , Genus `https://datos.iepnb.es/def/sector-publico/medio-ambiente/ifn/Genus` , Species `https://datos.iepnb.es/def/sector-publico/medio-ambiente/ifn/Species` , Hybrid `http://virtualforests.gsic.uva.es/Hybrid` , Subspecies `http://virtualforests.gsic.uva.es/Subspecies` , or variety `http://virtualforests.gsic.uva.es/Variety` .

Let's get all the species available in the dataset:

`https://crafts.gsic.uva.es/apis/virfor/query?id=taxa&type=https://datos.iepnb.es/def/sector-publico/medio-ambiente/ifn/Species`


GET https://crafts.gsic.uva.es/apis/virfor/query?id=taxa&type=https://datos.iepnb.es/def/sector-publico/medio-ambiente/ifn/Species

Params ● Authorization ● Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

	KEY	VALUE	DESCRIPTION
✓	id	taxa	
✓	type	https://datos.iepnb.es/def/sector-publico/medio-ambiente/ifn/Species	
	Key	Value	Description

Body Cookies Headers (10) Test Results — Status: 200 OK Time: 639 ms Size: 13.46 KB

```
        "bindings": [
            {
                "taxon": {
                    "type": "uri",
                    "value": "http://virtualforests.gsic.uva.es/SpeciesACERR_PSE"
                },
                "sciname": {
                    "type": "literal",
                    "xml:lang": "la",
                    "value": "Acer pseudoplatanus"
                },
                "esnames": {
                    "type": "literal",
                    "value": "Arce seudoplátano"
                },
                "ennames": {
                    "type": "literal",
                    "value": "Sycamore"
                },
                "frnames": {
                    "type": "literal",
                    "value": "Érable sycomore"
                }
            },
            {
                "taxon": {
                    "type": "uri",
                    "value": "http://virtualforests.gsic.uva.es/SpeciesBETUL_PEN"
```

Again, you can browse the results and get the IRIs of the species you are interested, e.g. `http://virtualforests.gsic.uva.es/SpeciesACERR_PSE` is the IRI of *Acer pseudoplatanus*.

8. The remaining template queries

We are now prepared to test the remaining template queries. Let's go with `provenances` (check the config file):

```
{
  "id": "provenances",
  "description": "Obtain all the provenances (variable \"provenance\") in the Virtual
     Forests repository, optionally filtered by taxon (variable \"taxon\")",
  "template": "select distinct ?provenance where {\n  ?provenance a <http://virtualforests.gsic.uva.
  "variables": [
     "provenance"
  ],
  "parameters": [
    {
      "label": "taxon",
      "type": "iri",
      "optional": true
    }
  ],
  "endpoint": "vf"
}
```

We can use this template to find all the provenances of species *Acer pseudoplatanus*:

`https://crafts.gsic.uva.es/apis/virfor/query?id=provenances&taxon=http://virtualforests.gsic.uva.es/SpeciesACERR_PSE`

The last query template is `arbProvInfos`:

```
{
   "id": "arbProvInfos",
   "description": "Obtain the set of arboreta provenance info objects (variable
      \"arbprovinfo\") in the Virtual Forests repository corresponding to a set
      of arboreta (compulsory parameter \"arboretum\"), with an optional provenance
      (parameter \"provenance\") or an optional taxon (parameter \"taxon\")",
   "template": "select distinct ?arbprovinfo ?arboretum where {\n  VALUES ?arboretum { {{#arboretum}}
   "variables": [
      "arbprovinfo",
      "arboretum"
   ],
   "parameters": [
      {
        "label": "arboretum",
        "type": "iri[]",
        "optional": false
      },
      {
        "label": "provenance",
        "type": "iri",
        "optional": true
      },
      {
        "label": "taxon",
        "type": "iri",
        "optional": true
      }
    ],
   "endpoint": "vf"
}
```

This is a flexible template for gathering arboreta provenance info objects. For example, we might be interested in finding the provenance info objects for provenance `http://reinfforce.iefc.net/data/provenance/ACERR_PSE-ALPS` in three arboreta (`http://reinfforce.iefc.net/data/arboretum/AR01`, `http://reinfforce.iefc.net/data/arboretum/AR02`, and `http://reinfforce.iefc.net/data/arboretum/AR03`):

`https://crafts.gsic.uva.es/apis/virfor/query?id=arbProvInfos&provenance=http://reinfforce.iefc.net/data/provenance/ACERR_PSE-ALPS&arboretum=http://reinfforce.iefc.net/data/arboretum/AR01&arboretum=http://reinfforce.iefc.net/data/arboretum/AR02&arboretum=http://reinfforce.iefc.net/data/arboretum/AR03`

THE END