RESTful Web Services

Table of Contents:

Understanding the URL

The general structure of a CellBase RESTful call is:

```
http://HOST_URL/{version}/{species}/{category}/{subcategory}/{id}/{resource}?{filters}
```

Where ***HOST_URL*** is

http://bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest

Sections in braces are parameters, so they can be treated as variables.

Example:

```
bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2/info
```

As is explained later in this documentation, this RESTful WS will get all the information available for the gene BRCA2 in human using the *v4* version.

The available parameters are:

Version

Indicate the CellBase version to retrieve information from. Versions are numbered as *v1*, *v2*, *v3*, *v4,* etc. At this moment the latest stable version is ***v4***.

Species

Species to get information from. A list of the species available for version v4 can be found at Data sources and species. Use the **Id** to indicate the species. For example, **hsapiens** for *Homo sapiens* or **mm usculus** for Mus musculus.

Category and subcategory

These parameters must be specified depending on the nature of your input data. For example, if we want to query CellBase by a genomic region (e.g. 13:32889575-32889763) we should use the *genomic* categor y and *region* subcategory. There are 4 main categories:

Category	Description	Subcategories
Genomic	Genomic category makes reference to genomic coordinates	Position, Variant, Region
Feature	Feature category involve all elements which have a defined location on the genome and provides an easy way to retrieve cross references for an ID	Gene, Variation, Transcript, Protein, Xref, ...
Regulatory	Regulatory category refers to all regulatory interactions involving transcription factors and microRNAs	TFBSs
Meta	Meta category provides resources to access CellBase meta-data, including available code versions, species or functioality	-

Id

What the user wants to retrieve from the id. This is the query parameter, it is the feature or term about we want to retrieve the information (**resource**). Its type must correspond with the **subcategory**.

> *NOTE: In order to improve performance, ID lists can be passed together in only one REST call separated by commas. Only 200 IDs are allowed. For larger queries user the CellBase clients.*

Examples:

- bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/genomic/region/3:1000-200000,X:35-459000,4:2334-5555/gene
- bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2,BCL2/snp?include=chromosome,start,end,id (returned data limited to start, end and rs id to reduce response time in this last query)

Resources

Each Category and Subcategory can have different resources and actions allowed. They specify the type of result we want to obtain from the ID.

Examples:

- bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2/transcript
- bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/genomic/region/3:1000-200000/gene
- bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/genomic/region/3:100000-200000/snp

> *NOTE: Resources must always be written in **singular**.*

Filters and extra options

Filters and extra options can be added at the end of the REST query followed by ?. They are optional and can be combined using the & sign. E.g:

bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2/info?of=json&include=id,name,biotype

These are the available filters and options:

output format: coded as *of**, the only allowed value now is *json* (default), others such as *protobuf* are being developed. E.g.:

bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2/info?of=json

exclude: name of the fields to be excluded in the output. E.g:

bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2/info?exclude=transcripts

include: name of the fields to be included in the output, the rest will be excluded. E.g:

bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2/info?include=id,name,biotype

limit: maximum number of results to be returned. By default, all results are returned. E.g:

bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2/snp?limit=3

skip: number of results to be skipped. By default, no result is omitted. E.g:

bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2/snp?skip=5

count: get the number of results obtained. By default, false. E.g:

bioinfo.hpc.cam.ac.uk/cellbase/webservices/rest/v4/hsapiens/feature/gene/BRCA2/snp?count=true

Results

API

A list of all available web services end points is available at:

http://bioinfo.hpc.cam.ac.uk/cellbase/webservices/

powered by the Swagger project.

Python client. PyCellBase

- This Python package makes use of the exhaustive RESTful Web service API that has been implemented for the CellBase database.
- It enables to query and obtain a wealth of biological information from a single database, saving a lot of time.
- As all information is integrated, queries about different biological topics can be easily and all this information can be linked together.
- Currently *Homo sapiens*, *Mus musculus* and a total of 32 species are available and many others will be included soon.
- More info about this package in the Python client tutorial of the CellBase Wiki

Package notes

- **PyCellBase** is compatible with both Python 2 and 3.
- This package makes use of multithreading to improve performance when the number of queries exceed a specific limit.
- It is distributed:
    - With the rest of the CellBase code at `cellbase/clients/python`.
    - Via *PyPI - the Python Package Index* **(not available yet)**

General usage

PyCellBase code can be accessed at `https://github.com/opencb/cellbase/tree/develop/clients/python/pycellbase`.

The **CellBaseClient** class provides access to the **different clients** of the data we want to query (e.g. gene, transcript, variation, protein, genomic region, variant).

Each of these clients provide a **set of methods** to ask for the resources we want to retrieve. Most of these methods will need to be provided with **comma-separated IDs or list of IDs**. Optional filters and extra options can be added as key-value parameters.

Responses are retrieved as **JSON formatted data**. Therefore, fields can be queried by key.

If there is an available resource, but there is not an available method in this python package, the **CellBaseClient** class can be used to **create the URL of interest**. This class is able to access the RESTful Web Services through the *get* method it implements. In this case, this method needs to be provided with those parameters which are required by the URL: category (e.g. feature), subcategory (e.g. gene), ID to search for (e.g. BRCA1) and method to query (e.g. search).

Configuration data as host, API version, or species is stored in a ConfigClient object. A custom configuration can be passed to CellBaseClient with a ConfigClient object provided with a JSON or YML config file. If you want to change the configuration on the fly you can directly modify the ConfigClient object.

Please, find more details on how to use the python library at: Python client tutorial

Installation

Cloning

PyCellBase can be cloned in your local machine by executing in your terminal:

```
$ git clone https://github.com/opencb/cellbase.git
```

Once you have downloaded the project you can install the library:

```
$ cd cellbase/clients/python
$ python setup.py install
```

PyPI

(not available yet)

R client

Note: R client library available from version 4 onwards

R library package is implemented and maintained for the latest R release and distributed through Bioconductor (https://bioconductor.org/packages/release/bioc/html/cellbaseR.html). For a quick install, please enter the R terminal and type:

```
source("https://bioconductor.org/biocLite.R")
biocLite("cellbaseR")
```

The R code is also distributed together with the rest of the CellBase code. R code can be found at: `cellbase/clients/R`

The R code provides a library for programmatic access to CellBase data. No CLI is implemented in R.

Comprehensive description of the library is provided by the Bioconductor documentation:

https://bioconductor.org/packages/release/bioc/html/cellbaseR.html