

Using the Scopus APIs

Documentation

This documentation provides an overview of how to work with the Scopus APIs, including a selection of sample PHP scripts for doing so.

CONTENTS

- [Overview](#)
- [Scopus API documentation](#)
- [Getting an API key](#)
- [Scopus Licensing](#)
- [API types](#)
- [Scripts](#)
- [Using the scripts](#)

OVERVIEW

This package includes a set of scripts that can be used to query the Scopus API (*application program interface*). Although these particular scripts are written in PHP (<http://www.php.net/>) with API calls executed via cURL, any other scripting language that enables interaction with a RESTful API would produce similar results. The following sections discuss the scripts themselves as well as the skills and materials that are assumed by the user before they can be utilized.

SCOPUS API DOCUMENTATION

Scopus provides comprehensive documentation for all of their available content APIs, including information about the types of data that can be retrieved and how to build query strings. This documentation is available at <http://api.elsevier.com/documentation/apis.html>.

GETTING AN API KEY

In order to use any of the Scopus APIs, you must first register for an API key. This can be done through the Elsevier Developers portal at <http://dev.elsevier.com/>.

Once you have registered for an API key, this key should be referenced in any script used to query the APIs; those locations are documented within the scripts themselves.

SCOPUS LICENSING

The use of Scopus, including the APIs, is dependent upon the terms and conditions of institutional licensing. Refer to your institutional contract to determine whether your usage of the APIs falls within your granted credentials.

API TYPES

The following APIs are utilized in the scripts included in this package:

API name	Script description and documentation
content/search/scopus	Performs a query against Scopus publication content and retrieves data on matching publications <i>Documentation</i> http://api.elsevier.com/documentation/SCOPUSSearchAPI.wadl
content/search/author	Performs a query against authors profiled in Scopus and retrieves data on matching author records <i>Documentation</i> http://api.elsevier.com/documentation/AUTHORSearchAPI.wadl

SCOPUS VARIABLES: DEFINITIONS

The Scopus APIs link data between them using a set of variables with specific names, described below:

Variable name	Variable description
Scopus author ID	<p>A unique, proprietary identifier that is assigned to all disambiguated authors profiled in Scopus; each ID points to a unique Scopus author profile at</p> <p>http://www.scopus.com/authid/detail.url?authorId=[Scopus author ID]</p> <p>Note: When distinct Scopus profiles are merged into one (such as when it is found that two profiles refer to the same individual), typically the associated Scopus author IDs referring to the previously unmerged profiles become aliased and automatically forward API calls to the Scopus author ID of the merged profile.</p>
Scopus publication ID	<p>A unique, proprietary identifier assigned to all unique publication records in Scopus, typically indicated in API returns by SCOPUS_ID.</p> <p>Note: These IDs can change or disappear over time.</p>
Scopus publication eID	<p>A unique <i>electronic</i> identifier that provides stable and persistent reference to a unique publication record in Scopus, typically derived from the Scopus publication ID</p>

SCRIPTS

The following scripts are included in this package:

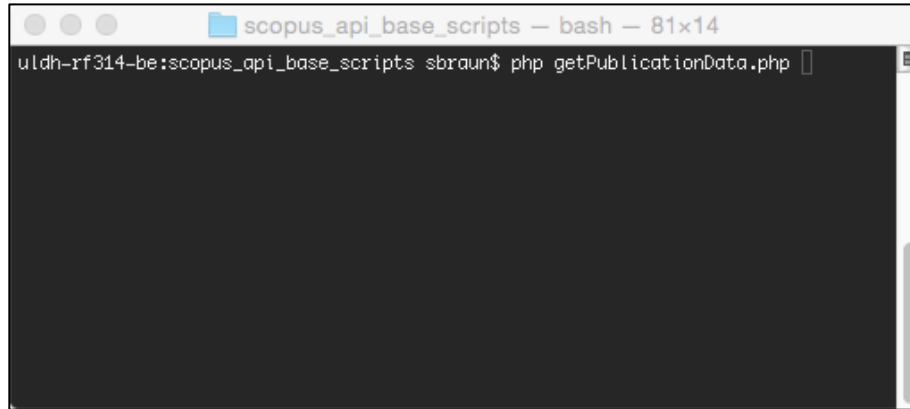
Script name	Base API	Script description
getAuthorPublications.php	content/search/scopus	Retrieves data about publications authored by a specified list of people, based on known Scopus author IDs
getPublicationData.php	content/search/scopus	Retrieves data about a specified list of publications, based on known Scopus publication eIDs (electronic identifiers)
searchForAuthor.php	content/search/author	Searches for and retrieves data about authors profiled in Scopus, based on name, affiliation, and/or other parameters

USING THE SCRIPTS

In the examples given, API queries are carried out through calls made via php cURL. Since these scripts are written in php, they may be run via any standard bash terminal.

For example, to run the script **getPublicationData.php** in terminal, navigate to the directory in which the file is located and execute the command

> **php getPublicationData.php**



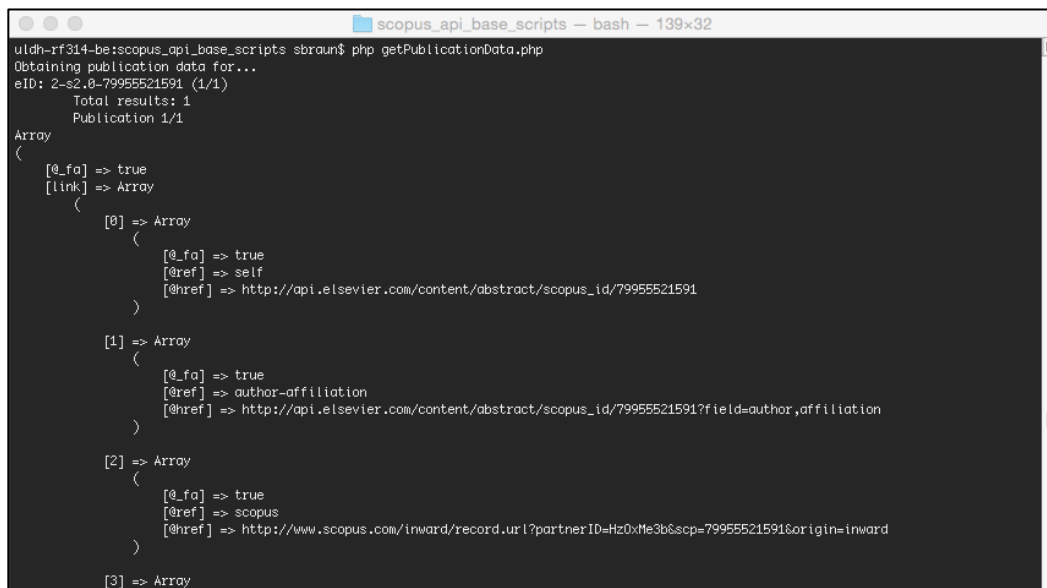
```

scopus_api_base_scripts — bash — 81x14
uldh-rf314-be:scopus_api_base_scripts sbraun$ php getPublicationData.php

```

USING THE DATA

The Scopus APIs can return data in different formats, including JSON or XML, depending on user specifications. Your choice of format will depend upon how you intend to use the data after they are retrieved. In the scripts included here, JSON is specified as the format of choice. The examples print out the retrieved API query results to the terminal screen; an example of what this looks like is shown below, using **getPublicationData.php** to retrieve data for the publication with known eID **2-s2.0-79955521591**.



```

scopus_api_base_scripts — bash — 139x32
uldh-rf314-be:scopus_api_base_scripts sbraun$ php getPublicationData.php
Obtaining publication data for...
eID: 2-s2.0-79955521591 (1/1)
Total results: 1
Publication 1/1
Array
(
    [@_fa] => true
    [link] => Array
        (
            [0] => Array
                (
                    [@_fa] => true
                    [@ref] => self
                    [@href] => http://api.elsevier.com/content/abstract/scopus_id/79955521591
                )
            [1] => Array
                (
                    [@_fa] => true
                    [@ref] => author-affiliation
                    [@href] => http://api.elsevier.com/content/abstract/scopus_id/79955521591?field=author,affiliation
                )
            [2] => Array
                (
                    [@_fa] => true
                    [@ref] => scopus
                    [@href] => http://www.scopus.com/inward/record.url?partnerID=Hz0xMe3b&scp=79955521591&origin=inward
                )
            [3] => Array

```

One option for working with the data is to inject data into a database for storage and analysis. A popular choice for this is MySQL (<https://www.mysql.com/>), for which php has a reliable and powerful API (<http://php.net/manual/en/book.mysql.php>) built in.