Sciwheel API

Contents

General API Information 2
  What our API can be used for 2
  User authorization 2
  API limitations 2
  Base URL 2
  General parameters and points of interest 2
    Pagination/Limitation of results 2
    HTTP status codes and Error handling 3
    Null output fields 3
API requests 3
  Requesting references in bulk 3
    Base HTTP GET request 3
    Query parameters 3
    Output fields 4
    Example reference object 6
    Example reference queries & responses 6
  Requesting a single reference 9
    Base HTTP GET request 9
    Example 9
  Generating reference metadata files 10
    Base HTTP GET request 10
    Query parameters 10
    Example 10
  Requesting notes 11
    Project notes: Base HTTP GET request 11
    Query parameters 11
    Output fields 11
    Example note object 12
    Example note query & response 12
  Requesting Projects 13
    Base HTTP GET request 13
    Query parameters 13
    Output fields 13
    Example project object 13
    Example project query & response 14
Contact details 15
General API Information

What our API can be used for
Our API allows a user to retrieve references, notes and projects which they have access to in Sciwheel (e.g. those they have added themselves or which have been shared with them) in JSON format. It is also possible to retrieve a list of your references in BibTeX, RIS or CSV format.

User authorization
Our API can only be used to retrieve data from user accounts for which you have a valid access token. An access token can be generated by any user by performing the following steps:

- Sign in to Sciwheel with the relevant user
- Go to My Account > External API
- Here you can view/copy your token or generate a new one (please note that generating a new token will invalidate any existing token for this user account)

This access token should be supplied in the API request as an HTTP header in the form:

```plaintext
Authorization: Bearer [Access token]
```  

API limitations
An access token can only be used to make 100 requests per 60 seconds; if more requests than this are made then an error message will be returned as detailed in Rate Limiting.

Base URL
The base URL for the Sciwheel API is:

https://sciwheel.com/extapi/work/

Please note that requests must use HTTPS.

General parameters and points of interest

Pagination/Limitation of results
Some requests returning multiple objects will return them in a paginated fashion. In such cases the full list of items will be returned containing pagination fields; the user can also specify their own page size, or choose to only return specific pages from the results. Examples of requests which include pagination parameters can be seen here, the details of the parameters are listed below:

https://sciwheel.com/extapi/work/references?page=3&size=25
https://sciwheel.com/extapi/work/references?size=2

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Mandatory</th>
<th>Values</th>
<th>Default</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>page</td>
<td>Integer</td>
<td>N</td>
<td>1,2,3...</td>
<td>1</td>
<td>Retrieve a specific page from the results (e.g. return page 3)</td>
</tr>
<tr>
<td>size</td>
<td>Integer</td>
<td>N</td>
<td>1 - 100</td>
<td>20</td>
<td>Number of results to include per page</td>
</tr>
</tbody>
</table>

Where a response is paginated, the results will return with the following JSON fields:
### Name Type Information

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>total</td>
<td>Integer</td>
<td>Total number of items matching query parameters</td>
</tr>
<tr>
<td>results</td>
<td>List</td>
<td>List of result items in the current page</td>
</tr>
<tr>
<td>page</td>
<td>Integer</td>
<td>Current page number</td>
</tr>
<tr>
<td>pageSize</td>
<td>Integer</td>
<td>Number of results returned per page (note, the results list may be smaller than this if fewer items are returned)</td>
</tr>
<tr>
<td>totalPages</td>
<td>Integer</td>
<td>Total number of pages as determined by pageSize</td>
</tr>
</tbody>
</table>

### HTTP status codes and Error handling

The API will respond to requests with one of the following codes:

<table>
<thead>
<tr>
<th>HTTP Code</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 Ok</td>
<td>Successful request</td>
</tr>
<tr>
<td>401 Unauthorized</td>
<td>User authorization error (e.g., invalid authorization token), or client has broken the rate limit</td>
</tr>
<tr>
<td>403 Forbidden</td>
<td>Insufficient privileges for the request</td>
</tr>
<tr>
<td>500 Internal Server Error</td>
<td>Server-related issue. Please contact us if the error persists</td>
</tr>
</tbody>
</table>

### Rate limiting

The Sciwheel API is currently rate-limited to prevent abuse. If too many requests are made in a given period, the API will return the error 401 Unauthorized. Clients receiving this error code should wait for 60 seconds and then try again.

### Null output fields

Please note that some fields in the JSON output (for example pubmedCitationsCount in a reference object) can be null values.

### API requests

#### Requesting references in bulk

**Base HTTP GET request**

```
https://sciwheel.com/extapi/work/references
```

This request will return metadata for references in the requesting user’s library (whether created by or shared with the user). If no further parameters are included then all references will be returned.

### Query parameters

Note that in addition to the following list, reference results are paginated.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Mandatory</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>String</td>
<td>N</td>
<td>Query terms which match any field in the reference</td>
</tr>
<tr>
<td>projectId</td>
<td>Integer</td>
<td>N</td>
<td>Query only for references inside this project. The user must be a member of queried projects</td>
</tr>
<tr>
<td>tagIds</td>
<td>Array of Integer</td>
<td>N</td>
<td>Return only references tagged with ALL given tag ids</td>
</tr>
<tr>
<td>bookmarked</td>
<td>Boolean</td>
<td>N</td>
<td>Return only references in the user’s Reading List</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Mandatory</td>
<td>Information</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>unsorted</td>
<td>Boolean</td>
<td>N</td>
<td>Return only references which are not in any project</td>
</tr>
<tr>
<td>incomplete</td>
<td>Boolean</td>
<td>N</td>
<td>Return only references with missing citation data (missing metadata details)</td>
</tr>
<tr>
<td>sort</td>
<td>Map&lt;Key,Value&gt;</td>
<td>N</td>
<td>Order results according to a set of fields of your choice - see Sort options (key-value pairs)</td>
</tr>
</tbody>
</table>

**Sort options (key-value pairs)**

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>relevance</td>
<td>asc,desc</td>
<td>Relevance of reference as determined by query parameters</td>
</tr>
<tr>
<td>title</td>
<td>asc,desc</td>
<td>Reference title</td>
</tr>
<tr>
<td>firstAuthor</td>
<td>asc,desc</td>
<td>First author (Last name)</td>
</tr>
<tr>
<td>lastAuthor</td>
<td>asc,desc</td>
<td>Last author (Last name)</td>
</tr>
<tr>
<td>publishedDate</td>
<td>asc,desc</td>
<td>Publication date</td>
</tr>
<tr>
<td>addedDate</td>
<td>asc,desc</td>
<td>Date the article was added to Sciwheel</td>
</tr>
<tr>
<td>journalName</td>
<td>asc,desc</td>
<td>Journal name</td>
</tr>
<tr>
<td>itemld</td>
<td>asc,desc</td>
<td>Reference ID (internal Sciwheel ID)</td>
</tr>
<tr>
<td>firstEditor</td>
<td>asc,desc</td>
<td>Book/series first editor (Last name)</td>
</tr>
<tr>
<td>chapter</td>
<td>asc,desc</td>
<td>Chapter title</td>
</tr>
<tr>
<td>url</td>
<td>asc,desc</td>
<td>Reference URL</td>
</tr>
<tr>
<td>itemType</td>
<td>asc,desc</td>
<td>Reference type (e.g. Article, Book, Thesis etc.)</td>
</tr>
<tr>
<td>publisher</td>
<td>asc,desc</td>
<td>Publisher name</td>
</tr>
<tr>
<td>edition</td>
<td>asc,desc</td>
<td>Edition (for supported types e.g. Book, Report)</td>
</tr>
<tr>
<td>publicationPlace</td>
<td>asc,desc</td>
<td>Place of publication (for supported types e.g. Book, Patent)</td>
</tr>
<tr>
<td>volume</td>
<td>asc,desc</td>
<td>Volume</td>
</tr>
<tr>
<td>issue</td>
<td>asc,desc</td>
<td>Issue number</td>
</tr>
<tr>
<td>pagination</td>
<td>asc,desc</td>
<td>Reference pagination (location in original publication, not to be confused with API result pagination)</td>
</tr>
<tr>
<td>doi</td>
<td>asc,desc</td>
<td>DOI ID</td>
</tr>
<tr>
<td>pubMedId</td>
<td>asc,desc</td>
<td>PubMed ID</td>
</tr>
<tr>
<td>patentNumber</td>
<td>asc,desc</td>
<td>Patent Number/ID</td>
</tr>
<tr>
<td>addedByName</td>
<td>asc,desc</td>
<td>Last name of user who added the reference to Sciwheel</td>
</tr>
<tr>
<td>hasPdf</td>
<td>asc,desc</td>
<td>Whether the reference has a PDF attached in Sciwheel</td>
</tr>
</tbody>
</table>

**Output fields**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
<td>Reference ID (internal Sciwheel ID)</td>
</tr>
<tr>
<td>type</td>
<td>String</td>
<td>See Available values for Type field</td>
</tr>
<tr>
<td>pubMedId</td>
<td>Integer</td>
<td>PubMed identifier</td>
</tr>
<tr>
<td>doi</td>
<td>String</td>
<td>DOI identifier</td>
</tr>
<tr>
<td>pmcId</td>
<td>String</td>
<td>PubMed Central identifier</td>
</tr>
<tr>
<td>title</td>
<td>String</td>
<td>Title</td>
</tr>
<tr>
<td>abstractText</td>
<td>String</td>
<td>Abstract text</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Information</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>publicationDate</td>
<td>String</td>
<td>String representation of publication date</td>
</tr>
<tr>
<td>publishedYear</td>
<td>Integer</td>
<td>Year of publication</td>
</tr>
<tr>
<td>volume</td>
<td>String</td>
<td>Volume</td>
</tr>
<tr>
<td>issue</td>
<td>String</td>
<td>Issue</td>
</tr>
<tr>
<td>pagination</td>
<td>String</td>
<td>Pages of issue reference appears on</td>
</tr>
<tr>
<td>journalName</td>
<td>String</td>
<td>Journal name</td>
</tr>
<tr>
<td>journalAbbreviation</td>
<td>String</td>
<td>Journal abbreviation</td>
</tr>
<tr>
<td>authorsText</td>
<td>String</td>
<td>Author names, comma-separated (e.g. Smith RA, Doe J)</td>
</tr>
<tr>
<td>fullTextLink</td>
<td>String</td>
<td>URL of full text on web</td>
</tr>
<tr>
<td>pdfUrl</td>
<td>String</td>
<td>URL of PDF in Sciwheel</td>
</tr>
<tr>
<td>pdfSize</td>
<td>Long</td>
<td>PDF file size (bytes)</td>
</tr>
<tr>
<td>f1000Recommended</td>
<td>Boolean</td>
<td>Article recommended in Faculty Opinions</td>
</tr>
<tr>
<td>f1000Bookmarked</td>
<td>Boolean</td>
<td>Reference is in the user's Reading List</td>
</tr>
<tr>
<td>f1000Incomplete</td>
<td>Boolean</td>
<td>Reference has missing citation data</td>
</tr>
<tr>
<td>f1000NotesCount</td>
<td>Integer</td>
<td>Number of notes by current user/collaborators</td>
</tr>
<tr>
<td>f1000AddedBy</td>
<td>String</td>
<td>User who added this reference to Sciwheel</td>
</tr>
<tr>
<td>f1000AddedDate</td>
<td>Date</td>
<td>Date reference added to Sciwheel (time in milliseconds since Jan 1, 1970</td>
</tr>
<tr>
<td></td>
<td></td>
<td>00:00:00 UTC)</td>
</tr>
<tr>
<td>f1000RecommendationsCount</td>
<td>Integer</td>
<td>Number of times article recommended in Faculty Opinions</td>
</tr>
<tr>
<td>f1000Tags</td>
<td>List&lt;String&gt;</td>
<td>Tags applied to the reference</td>
</tr>
<tr>
<td>pubmedCitationsCount</td>
<td>Integer</td>
<td>Number of times article was cited according to Europe PMC</td>
</tr>
</tbody>
</table>

**Available values for Type field**

<table>
<thead>
<tr>
<th>Type</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOG_POST</td>
<td>Social media Post</td>
</tr>
<tr>
<td>BOOK</td>
<td>Book</td>
</tr>
<tr>
<td>BROADCAST</td>
<td>Broadcast</td>
</tr>
<tr>
<td>CONFERENCE_PAPER</td>
<td>Conference paper</td>
</tr>
<tr>
<td>F1000_ARTICLE</td>
<td>Journal Article (metadata is partially curated by Sciwheel)</td>
</tr>
<tr>
<td>FILM</td>
<td>Film</td>
</tr>
<tr>
<td>FORUM_POST</td>
<td>Forum post</td>
</tr>
<tr>
<td>LEGAL_CASE</td>
<td>Legal case</td>
</tr>
<tr>
<td>MAGAZINE_ARTICLE</td>
<td>Magazine article</td>
</tr>
<tr>
<td>MANUAL_ITEM_PDF</td>
<td>PDF</td>
</tr>
<tr>
<td>MANUSCRIPT</td>
<td>Manuscript (where uploaded to an Sciwheel project)</td>
</tr>
<tr>
<td>MAP</td>
<td>Map</td>
</tr>
<tr>
<td>NEWSPAPER_ARTICLE</td>
<td>Newspaper article</td>
</tr>
<tr>
<td>OTHER</td>
<td>Other</td>
</tr>
<tr>
<td>PATENT</td>
<td>Patent</td>
</tr>
<tr>
<td>PERSONAL_COMMUNICATION</td>
<td>Personal communication</td>
</tr>
<tr>
<td>PRESENTATION</td>
<td>Presentation</td>
</tr>
<tr>
<td>REPORT_PAPER</td>
<td>Report paper</td>
</tr>
</tbody>
</table>
Zygomycetes in human disease.

The Zygomycetes represent relatively uncommon isolates in the clinical laboratory, reflecting either environmental contaminants or, less commonly, a clinical disease called zygomycosis. There are two orders of Zygomycetes containing organisms that cause human disease, the Mucorales and the Entomophthorales.

"publicationDate": "2000 Apr",
"publishedYear": 2000,
"volume": "13",
"issue": "2",
"pagination": "236-301",
"journalName": "Clinical Microbiology Reviews",
"journalAbbreviation": "Clin Microbiol Rev",
"authorsText": "Ribes JA, Vanover-Sams CL, Baker DJ",
"pdfUrl": null,
"pdfSize": null,
"f1000Recommended": false,
"f1000Bookmarked": false,
"f1000Incomplete": false,
"f1000NotesCount": 0,
"f1000AddedBy": "Katherine Kalinina",
"f1000AddedDate": 1495354383561,
"f1000RecommendationsCount": 0,
"f1000Tags": [],
"pubmedCitationsCount": null
}
Zygomycetes in human disease.

The Zygomycetes represent relatively uncommon isolates in the clinical laboratory, reflecting either environmental contaminants or, less commonly, a clinical disease called zygomycosis.

Zoonotic helminth infections with particular emphasis on fasciolosis and other trematodiasis.

Zoonotic infections are among the most common on earth and are responsible for >60 per cent of all human infectious diseases.
Query 2: References in a project with two tags applied

https://sciwheel.com/extapi/work/references?projectId=336430&tagIds=468&tagIds=1215

Response 2

```json
{
  "total": 1,
  "results": [
    {
      "id": "930395",
      "type": "RESEARCH_ARTICLE",
      "pubMedId": "28766364",
      "doi": "10.4155/bio-2017-0102",
      "pmcId": null,
      "title": "Overcoming disease-specific matrix effect in a clinical pharmacokinetic assay using a microfluidic immunoassay technology."
    }
  ]
}
```

The reference is:

- **Title**: Overcoming disease-specific matrix effect in a clinical pharmacokinetic assay using a microfluidic immunoassay technology.
- **Authors**: Williams K, Erickson R, Fischer SK
- **DOI**: 10.4155/bio-2017-0102
- **Journal**: Bioanalysis
- **Publication Date**: 2017 Aug 02
- **Published Year**: 2017

This reference is related to the project with two tags applied.
Requesting a single reference

Base HTTP GET request

```
https://sciwheel.com/extapi/work/references/{id}
```

This request will return metadata for the reference with id number {id} in the requesting user's library (whether created by or shared with the user).

Example

Query: Return metadata for reference ID 933202

```
https://sciwheel.com/extapi/work/references/933202/
```

Response

```json
{
    "id": "933202",
    "type": "RESEARCH_ARTICLE",
    "pubMedId": "30024914",
    "doi": "10.1371/journal.pone.0200455",
    "pmcId": "PMC6053154",
    "title": "Multiqubit and multilevel quantum reinforcement learning with quantum technologies.",
    "abstractText": "We propose a protocol to perform quantum reinforcement learning with quantum technologies. At variance with recent results on quantum reinforcement learning with superconducting circuits, in our current protocol coherent feedback during the learning process is not required, enabling its implementation in a wide variety of quantum systems. We consider diverse possible scenarios for an agent, an environment, and a register that connects them, involving multiqubit and multilevel systems, as well as open-system dynamics. We finally propose possible implementations of this protocol in trapped
```
ions and superconducting circuits. The field of quantum reinforcement learning with quantum technologies will enable enhanced quantum control, as well as more efficient machine learning calculations.

"publicationDate": "2018 Jul 19",
"publishedYear": 2018,
"volume": "13",
"issue": "7",
"pagination": "e0200455",
"journalName": "Plos One",
"journalAbbreviation": "PLoS ONE",
"authorsText": "Cárdenas-López FA, Lamata L, Retamal JC, Solano E",
"fullTextLink": "https://sciwheel.com/fulltext/doi/10.1371/journal.pone.0200455",
"pdfUrl": null,
"pdfSize": "2356610",
"f1000Recommended": false,
"f1000Bookmarked": false,
"f1000Incomplete": false,
"f1000NotesCount": 1,
"f1000AddedBy": "Yeshayahu Leibowitz",
"f1000AddedDate": 1533109436268,
"f1000RecommendationsCount": 0,
"f1000Tags": [],
"pubmedCitationsCount": "0"

Generating reference metadata files

Base HTTP GET request

**https://sciwheel.com/extapi/work/references/export**

This request will generate metadata for references in one of the following formats - BibTeX, RIS, CSV. A maximum of 10,000 references can be generated in a single request.

**Query parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Values</th>
<th>Default</th>
<th>Mandatory</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>projectId</td>
<td>Integer</td>
<td>1,2,3...</td>
<td>-</td>
<td>N</td>
<td>Generate metadata for all references inside this project. If none is supplied, all items in the user's library will be exported</td>
</tr>
<tr>
<td>exportType</td>
<td>Enum</td>
<td>BIBTEX, RIS, CSV</td>
<td>BIBTEX</td>
<td>N</td>
<td>Desired export format (if none is supplied then a BibTeX file will be returned). Note that the values are case-sensitive</td>
</tr>
</tbody>
</table>

**Example**

Query: Generate metadata for all references inside project with id 443377 in BibTeX format

**https://sciwheel.com/extapi/work/references/export?projectId=443377**
Response

@article{crdenaslpez_2018,
title = {Multiqubit and multilevel quantum reinforcement learning with quantum technologies.},
author = {Cárdenas-López, F A and Lamata, L and Retamal, J C and Solano, E},
pages = {e0200455},
url = {http://dx.doi.org/10.1371/journal.pone.0200455},
year = {2018},
month = {jul},
day = {19},
urldate = {2018-08-01},
journal = {{PLoS} {ONE}},
volume = {13},
number = {7},
doi = {10.1371/journal.pone.0200455},
pmid = {30024914},
pmcid = {PMC6053154},
f1000-projects = {collection1},
abstract = {We propose a protocol to perform quantum reinforcement learning with quantum technologies. At variance with recent results on quantum reinforcement learning with superconducting circuits, in our current protocol coherent feedback during the learning process is not required, enabling its implementation in a wide variety of quantum systems. We consider diverse possible scenarios for an agent, an environment, and a register that connects them, involving multiqubit and multilevel systems, as well as open-system dynamics. We finally propose possible implementations of this protocol in trapped ions and superconducting circuits. The field of quantum reinforcement learning with quantum technologies will enable enhanced quantum control, as well as more efficient machine learning calculations.}
}

Requesting notes

Notes can be requested either for a given reference {id} (this refers to the Sciwheel reference ID) or a given project {id} (this refers to the Sciwheel project ID), made by either the user or other members they are sharing the reference/project with.

Project notes: Base HTTP GET request

https://sciwheel.com/extapi/work/references/{id}/notes

This request will return a list of all notes made on the entered reference {id} (this refers to the Sciwheel reference id), made either by the user or other members of the shared project.

Query parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Value</th>
<th>Mandatory</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>sortingOrder</td>
<td>Enum</td>
<td>asc,desc</td>
<td>N</td>
<td>Sort by date each note was last updated</td>
</tr>
</tbody>
</table>

Output fields
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
<td>Note ID (Internal Sciwheel ID)</td>
</tr>
<tr>
<td>user</td>
<td>String</td>
<td>User who made the note (e.g. Joe Johnson)</td>
</tr>
<tr>
<td>comment</td>
<td>String</td>
<td>Note comment content</td>
</tr>
<tr>
<td>highlightText</td>
<td>String</td>
<td>Text highlighted as part of note using the Sciwheel browser extension (where applicable)</td>
</tr>
<tr>
<td>replies</td>
<td>List</td>
<td>List of replies to note (each reply is a note object)</td>
</tr>
<tr>
<td>created</td>
<td>Long</td>
<td>Number of seconds passed since the note was created (at time response was generated)</td>
</tr>
<tr>
<td>updated</td>
<td>Long</td>
<td>Number of seconds passed since the note was last updated (at time response was generated)</td>
</tr>
<tr>
<td>url</td>
<td>String</td>
<td>URL of highlighted reference/PDF file (where applicable)</td>
</tr>
</tbody>
</table>

**Example note object**

```json
{
    "id": "31498",
    "user": "Joe Johnson",
    "comment": "We should think about the meta-analysis.",
    "highlightText": "Anxiety disorders",
    "replies": [
    ],
    "created": 15504,
    "updated": 8947,
}
```

**Example note query & response**

*Query: Return all notes for reference ID 933202*

**Response**

```
[
    {
        "id": "31498",
        "user": "Katya Gorodinsky",
        "comment": "This is very important",
        "highlightText": "Anxiety disorders",
        "replies": [
        ],
        "created": 15857,
        "updated": 9300,
    },
    {
        "id": "31496",
        "user": "John Spencer",
        "comment": "We should check what are the implications",
        "highlightText": null,
        "replies": [
        
        ]
    },
    {
        "id": "31497",
```
Requesting Projects

Base HTTP GET request

https://sciwheel.com/extapi/work/projects

This request will return a list of all projects the user is a member of (whether created by or shared with them).

Query parameters

Projects returned by the query are paginated.

Output fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
<td>Project ID (Internal Sciwheel ID)</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Project name</td>
</tr>
<tr>
<td>children</td>
<td>List</td>
<td>Subprojects (each subproject is a project object)</td>
</tr>
<tr>
<td>parentId</td>
<td>Integer</td>
<td>ID of the parent project (where the current project is a subproject)</td>
</tr>
</tbody>
</table>

Example project object

```json
{
  "id": "318677",
  "name": "Hebrew citation in WP",
  "children": [
    {
      "id": "319349",
      "name": "article",
      "children": [],
      "parentId": "318677"
    },
    {
      "id": "319348",
      "name": "book",
      "children": [],
      "parentId": "318677"
    }
  ]
}
```
Example project query & response

Query: Return list of projects, 5 per page, only displaying page 3


Response

```json
{
  "total": 23,
  "results": [
    {
      "id": "338028",
      "name": "Hebrew citation in WP",
      "children": [],
      "parentId": null
    },
    {
      "id": "329464",
      "name": "Climate research",
      "children": [],
      "parentId": null
    },
    {
      "id": "338026",
      "name": "Lung Cancer",
      "children": [],
      "parentId": null
    },
    {
      "id": "337503",
      "name": "Biology Research Group",
      "children": [
        {
          "id": "337504",
          "name": "Alpha",
          "children": [],
          "parentId": "337503"
        }
      ],
      "parentId": null
    },
    {
      "id": "329460",
      "name": "Environmental Changes",
      "children": [],
      "parentId": null
    }
  ]
}
```
Contact details

If you have any problems or questions, please do feel free to contact us via the Sciwheel site, or by emailing feedback@sciwheel.com.