bio.tools Documentation

Release latest

Dec 18, 2017

Contents

This is the documentation for bio.tools.

Contents:

CHAPTER 1

What is bio.tools?

bio.tools is a portal to bioinformatics resources worldwide, aimed to help bioinformaticians and scientists:

- find, understand, compare and select resources == **discovery**
- use and connect them in workflows == **(inter)operability**

1.1 Objectives

Our main objectives are:

- build and maintain a comprehensive **registry** of high-quality software metadata / descriptions
- provide a **web portal** enabling registration, editing, search and discovery of the registry content
- support a **community** for the sustainable maintenance of the registry content and development of the portal features
- expose results of tool performance **benchmarking**, online service **monitoring** and other metrics of software and service quality
- integrate the registry with popular **workbench environments** in a way that improves resource interoperability
- **support** registry stakeholders including tool providers and end-users

1.2 Scope

bio.tools scope is *application software* with well-defined data processing functions (inputs, outputs and operations). bio.tools includes a broad range of software types including tools available for immediate use as online services, or in a form which which you can download, install, configure and run yourself. This includes simple tools with one or a few closely related functions, and complex, multimodal tools with many functions.

1.3 Technical components

- biotoolsSchema is a description model for bioinformatics software. It is a formalised XML schema (XSD) which defines 50 important scientific, technical and administrative attributes. It defines what attributes may be specified in a bio.tools entry, a precise syntax for those descriptions, and controlled vocabularies for consistent description of technical aspects such as software license and software type.
- EDAM ontology is an ontology of well-established, familiar concepts that are prevalent within bioinformatics and computational biology, including types of data and data identifiers, data formats, operations and topics. It defines precise semantics for the scientific description of software registered in bio.tools.
- Curation guidelines describe how each attribute should be specified, *i.e.* concern the quality of an entry. The guidelines go beyond the syntactic and semantic constraints defined by biotoolsSchema and EDAM, and are part of broader tool information standards being adopted by bio.tools.
- **Tool Cards** *e.g.* https://bio.tools/signalp provide key information at a glance for registered tools. Tool cards have human-friendly, persistent URLs which include the unique tool identifier ("signalp" in this case). The identifier is assigned upon registration is a URL-safe derivative of (normally identical to) the supplied tool name.
- **Query interfaces** available at https://bio.tools help bio.tools end-users with tool discovery and include the search bar, a compact "mini-card" view and a detailed "grid" view. See the Users Guide.
- **Registration interface** enables manual creation of valid registry content and editing, including graphical editing via tabbed panes and an interactive JSON editor with inline error reporting. It is available to logged-on users via "Menu ... Add content". See the Users Guide.
- bio.tools API provides programmatic means to query, add and edit registry content.
- **bio.tools metrics** available at https://bio.tools/stats include registry growth, contributors, annotation breakdown *etc*.

1.4 Docs overview

- Contributors Guide - how to get involved (please do!)
- User Guide - how to use the bio.tools user interfaces.
- Curators Guide - how to create a high quality bio.tools entry.
- Editors Guide - thematic editorships to improve bio.tools in scientific areas.
- Documentors Guide - how to edit the bio.tools docs.
- API reference - bio.tools API docs.
- Hangouts - monthly coordination meetings (you're welcome to join!)
- Roadmap - technical plans for the next year
- Studentships - bio.tools studentship scheme for curation-focussed mini-projects.
- GitHub projects - open projects of relevance to bio.tools.

bio.tools development is supported by ELIXIR - the European Research Infrastructure for life science information. bio.tools content is freely available to all under open license.

1.5 Getting involved : a quick-start guide

1. Read the docs but especially the contributors guide.
2. All project / high-level task management is done in sifter, email Jon Ison if you want in!
3. Join the mailing lists but note that most of the discussion is done via sifter. Important announcements and some discussion are done via registry-core (see below)
4. Members of registry-core share a mailing list and calendar, but there are some implications of joining. Email Jon if you want to join.
5. GitHub is used heavily for public / fine-grained issue tracking, code *etc.*, see the bio.tools and EDAM organisations, in particuar the biotoolsregistry and edamontology projects. Email Jon if you want to join.
6. We run monthly hangouts (coordination meetings) and - for technical people routinely involved with bio.tools curation or software development - weekely technical calls. To join the hangouts email Henriette cc Jon or to join the weekly calls email Emil cc Jon.
7. Dive in at the deep end! There are no end of projects and tasks to get involved with, see sifter and email Jon in the 1st instance to get orientated.

CHAPTER 2

Contributors Guide

2.1 Task management

All high-level tasks concering bio.tools are managed in **sifterapp** (moderated by Jon Ison): - https://biotools.sifterapp.com

If you'd like a sifterapp account, please email Jon).

Note: sifterapp is the primary means for technical coordination: collaborators are encouraged to browse tasks, review priorities, make comments and add new tasks. Bear in mind sifterapp is for *high level* tasks not fine-grained details!

2.2 bio.tools community site

GitHub is used for sharing code and data for all bio.tools-related projects:

- https://github.com/bio-tools/
- https://github.com/bio-tools/biotoolsschema
- https://github.com/bio-tools/biotoolsregistry
- https://github.com/edamontology/edamontology

2.3 Feature requests & issues

GitHub is used to track **fine-grained issues** and is the preferred way to make bio.tools feature requests, content suggestions, EDAM term requests, and bug reports:

- https://github.com/bio-tools/biotoolsregistry/issues
- https://github.com/bio-tools/biotoolsschema/issues

- https://github.com/edamontology/edamontology/issues

2.4 Mailing list

Please use the appropriate mailing list:

- **registry** for general discussions on bio.tools
- **registry-support** for questions and help on bio.tools
- **edam** for general discusion and help on EDAM
- **registry-announce** or **edam-announce** for low-traffic announcements
- **registry-core** and **edam-dev** for technical discussion amongst the core developers

To send mail: - registry@elixir-dk.org - registry-support@elixir-dk.org - edam@elixir-dk.org

2.5 Hangouts

Coordination meetings happen on the **last Fri of each month @ 11AM CET**. The hangouts have an open agenda and respond to current critical needs. Technical representatives of ELIXIR-DK institutes routinely attend and guests are very welcome: if you'd like to join mail Henriette Husum Bak-Jensen cc Jon Ison.

CHAPTER 3

User Guide

This user guide aims to help you through the different steps to register your own entry to bio.tools.

Note: If you find a bug, have any questions or suggestions, please get in touch with us.

3.1 Create an account

Creating an account on bio.tools is very quick and simple. Just click on the sign up button at the top-right corner of the page. Then you just need to give a username, your email address and a password to get your account done.

3.2 Add content

Every user is welcome to register its own resource to bio.tools. Once your account is created, you can start adding your content by clicking on add content.

The registration of an new entry is splitted in different parts that are described below.

Note: The minimum information required is marked with a red asterix *.

At every moment, you can check the validity of your information by clicking on Validate and save it by clicking on Save 

.

Note: Saving the entry makes it directly available online. If you want to save what you have done without publishing it, the only way is to go to the *JSON* part and save the .json file locally.

3.2.1 Summary

For this first part, you give the main descriptors of your entry. This includess the **name** of your resource with a **description**, its **version** and a **homepage URL**. A unique **ID** is automatically generated from the name but it is still possible to change it.

Note: A unique URL on bio.tools will be generated for the entry with the following format: http://bio.tools/tool/ID/version/version_number.

3.2.2 Function

This is where you describe the functionnality of your entry based on the EDAM ontology[1]. The description is built on the following diagram:

In each box, you can add as many fields as you want. You can also add a general comment about the function (*this is particularly useful when your entry has several functions*).

Note: It can be difficult to find the right ontology to describe your function(s), input(s) or output(s). Please visit OLS EDAM and BioPortal EDAM websites to find more information about the different ontologies and help you finding the best description.

3.2.3 Labels

In this part, you can tell more about your tool:

- What **type** of tool it is (command line tool, library. . .).
- In which **topic(s)** the tool belongs to (based on EDAM ontology[1]).
- In which **operating system** it is possible to use it.
- The **language** used to develop the tool, its **license** and **maturity**.
- The **accessibility** of your tool and its **cost**.

3.2.4 Contact

At least one contact is required to register a tool but you can add as many contact as you want.

[1] Ison, J., Kalaš, M., Jonassen, I., Bolser, D., Uludag, M., McWilliam, H., Malone, J., Lopez, R., Pettifer, S. and Rice, P. (2013). EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats. Bioinformatics, 29(10): 1325-1332.

Note: If you wish to mention people that participated in the development of the tool, you can use the *Credits* part.

3.2.5 Links

It is the place where your share links that do not belong to the other parts. For instance, it can be link to a mailing list, mirror or repository.

3.2.6 Download

You can here share all the different download links you want. It can be many different kind such as binaries, source code, biological data, test data (full list available on the drop down menu of **Download type**).

3.2.7 Documentation

Make your different documentations for your tool available here.

3.2.8 Publications

Share the different publications of the tool which can be the primary publication but also review or secondary references that are relevant to this tool. You can use either the **PubMed Central ID** (PMCID), the **PubMed ID** (PMID) or the **Digital Object ID** (DOI).

3.2.9 Credits

Credits represent all type of entities that participated in the tool. It can be a people, but also an institution or a consortium.

3.2.10 JSON

This is all the information you gave about your tool, formatted in JSON format.

3.2.11 Permissions

You can decide to make the entry either editable only by yourself, a list of users or anyone.

3.3 Update a resource

There is two way to update a resource from its tool card:

- Click on update this record if it concerns minor changes
- Click on create a new version to register a new version. This allows to keep all the information concerning the previous version

3.4 Remove a resource

From the tool card, click on update this record. Then you can remove the entry by clicking on the remove button Remove.

> **Warning:** Removing an entry is definitive.

3.5 Search for a tool

Coming soon...

3.6 References

CHAPTER 4

Curators Guide

Attention: UNDER CONSTRUCTION

- guidelines for bio.tools curators, including EDAM annotation guidelines.
- to make suggestions about these guidelines please add comments via GitHub
- for curation advice mail registry-support

bio.tools includes all types of bioinformatics *tools* - application software with well-defined data processing functions (inputs, outputs and operations). This ranges from simple tools with a single primary function, to complex, multimodal tools with many disinct functions. Tools may be available for immediate use as online services, or in a form which you can download, install, configure and run yourself.

Usually, a bio.tools entry describes a discrete tool. Some entries describe *collections* of tools, such as software suites. The scope, *i.e.* the types of tools that may be included, and the attributes for their description, are defined in biotoolsSchema which uses the EDAM ontology as a source of terms for the tool scientific description. These curation guidelines describe how to create a high quality tool description, above and beyond the syntactic and semantic constraints that are defined in biotoolsSchema and EDAM.

- general guidelines include basic considerations, annotation of tool functions and the use of EDAM. You should read these first of all.
- guidelines on specific attributes defined in the biotoolsSchema
- guidelines specific to individual types of tools

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119:

- **“MUST”**, **“REQUIRED”** or **“SHALL”** mean that the guideline is an absolute requirement of the specification.
- **“MUST NOT”** or **“SHALL NOT”** mean that the guideline is an absolute prohibition of the specification.
- **“SHOULD”** or **“RECOMMENDED”** mean that there may exist valid reasons in particular circumstances to ignore a particular guideline, but the full implications must be understood and carefully weighed before doing so.

- **"SHOULD NOT"** or the phrase **"NOT RECOMMENDED"** mean that there may exist valid reasons in particular circumstances when acting contrary to the geuideline is acceptable or even useful, but the full implications should be understood and the case carefully weighed before doing so.
- **"MAY** or **"OPTIONAL"** mean that the guideline is truly optional; you can choose to follow it or not.

Note: The guidelines are a key component of an emerging information standard for tools being adopted by bio.tools, as a basis to monitor content and label bio.tools entries.

- **automatically verified** guidelines are (or will be) checked *via* automated periodic QC of the bio.tools system
- **manually verified** guidelines are checked *via* manual QC performed by trusted curators (bio.tools admin, entry owners *etc.*)
- advice given in boxes (notes, tips, caution *etc.* are not verified

---

4.1 General guidelines

4.1.1 Before you start

Consider the following *before* creating a bio.tools entry:

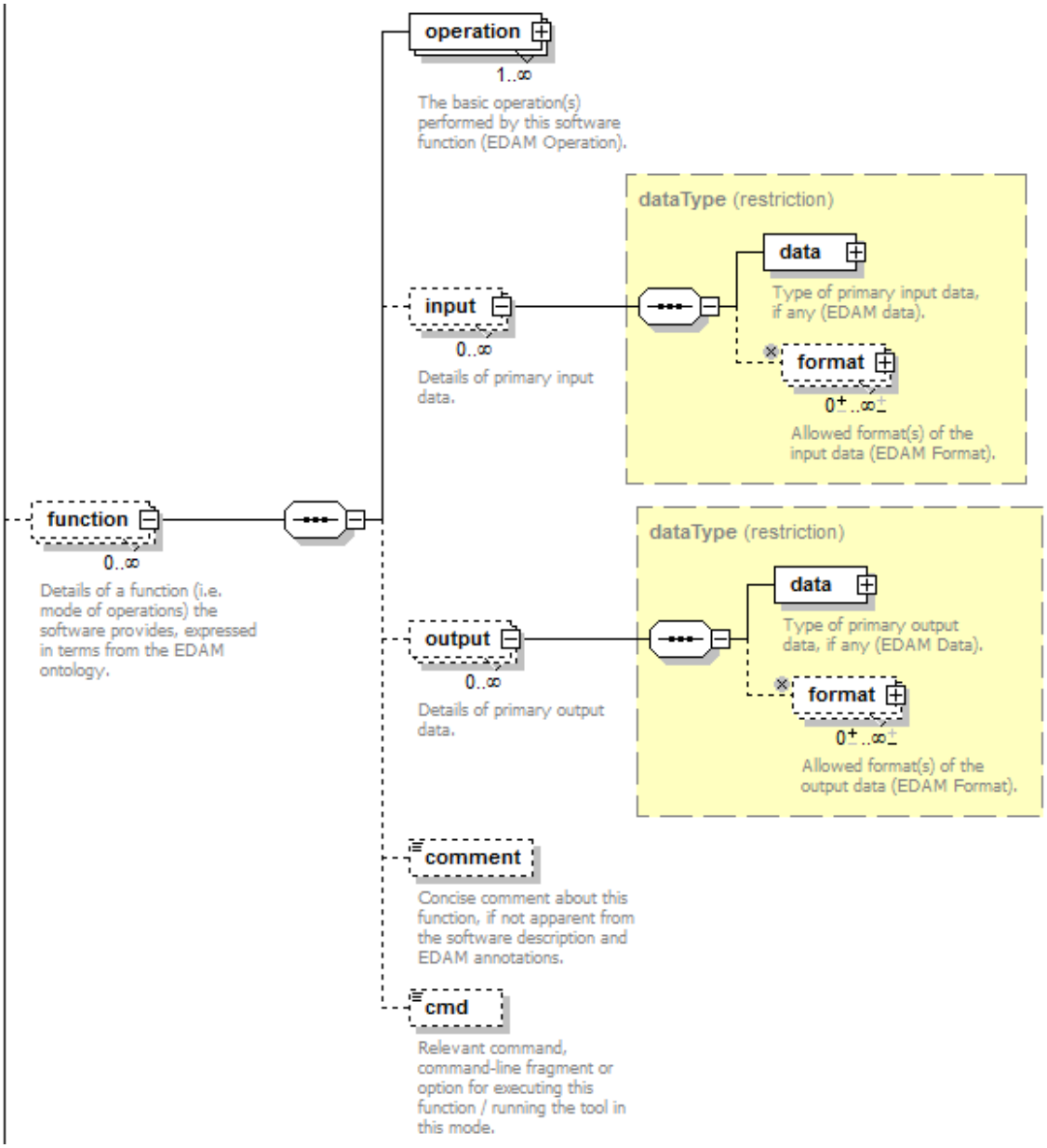
1. **Are one or more entries required to describe the software?**
   - workbenches and other suites often require multiple entries.
   - tools with multiple interfaces (*e.g.* Command-line tool , Web API, Web service and Web application) **SHOULD** be described by a single entry **unless** these interfaces provide fundamental functional differences (see Tool functions below).
   - if in doubt, mail registry-support.
2. **What tool types apply?**
   - one or more tool types may be assigned in a single entry reflecting different facets of the software described by the entry.
   - read the tool type-specific guidelines before you create the tool.
3. **What if the software is already registered?**
   - if you're the rightful owner of the entry (*i.e.* the tool developer or provider of an online service) then request ownership of it
   - otherwise, request edit rights

   Make these requests using the buttons at the bottom of the Tool Card (see *e.g.* https://bio.tools/signalp).

   If you plan to register multiple entries *en masse*, please discuss this first with bio.tools admin.
4. **Are there version-specific considerations?**
   - as a rule, a bio.tools entry **SHOULD** describe the *latest version* available at the time of registration and **SHOULD** be updated, as required, for subsequent releases.
   - if a new version has fundamental functional differences (see Tool functions below) it **MAY** be registered as an entirely new entry. In such cases, follow carefully the guidelines for tool name and version annotations.
5. **Plan** how to describe the tool functions.
6. **Read** the general EDAM annotations guidelines.

4.1.2 Tool functions

bio.tools uses a model of software (see below) defined within biotoolsSchema. A tool can have one or more basic functions (modes of operation), each function performing one or more specific operation (e.g."Sequence alignment"), each of which may have one or more primary inputs and outputs, each of a defined type of data and listing supported format(s).



Plan how how to describe the software:

- identify the distinct functions (modes of operation) and the individual operations associated with each one. Typically different functions (modes) perform different operations and for well documented tools, this is usually obvious. If in any doubt mail registry-support.
- as a general rule, if the tool allows an option between doing one thing or another, then you **MUST** annotate the operations as distinct functions. If in contrast a tool always does one or more things, then you **MUST** annotate these as distinct operations within a single function.

- bio.tools aims for fairly coarse-grained description, *i.e.* you **SHOULD** only specify the primary functions and operations, from a typical end-user perspective. If a tool happens to perform some operation internally, but this is secondary to its advertised purpose, then you **SHOULD NOT** annotate it. If in doubt mail registry-support
- this holds for input and output too, *e.g.* a sequence alignment tool would be annotated as reading sequences (input), and writing a sequence alignment (output), but not with gap insertion and extension penalties, or other parameters.
- many tools allow a primary input or output to be specified in a number of alternative ways, *e.g.* a sequence input that may be specified *via* a sequence identifier, or as a literal sequence. In such cases, you **MAY** annotate the alternatives as distinct functions (see above). If specifying just one alternative, you **SHOULD** use the EDAM Data concept for the type of data, rather than identifier.

Note: A future refactoring may improve the modelling for alternative specification of inputs and outputs, by allowing multiple data+format couplets for a given input or output. If this is done, the proposed guideline would be:

- you **MAY** annotate all the commonly used alternatives and, if specifying alternatives, **MUST** annotate these as distinct data + format couplets within a single input or output.
- many inputs and outputs are complex, with individual data files containining multiple types of data. You **MUST** select the single EDAM Data term that best describes an input or output (see EDAM annotations below) and **MUST NOT** specify multiple EDAM Data terms describing differents facets of the data.

Input on this issue is welcomed via GitHub.

4.1.3 EDAM annotations

The EDAM ontology is used to annotate applicable topics, operations, and the type and format of inputs and outputs. The general guidelines below apply for all EDAM annotations.

- **1. MUST NOT** use "organisational" EDAM concepts *e.g.* Topic of "Topic" or Operation of "Operation" (see note below)
- **2. SHOULD** use the most specific term(s) available, bearing in mind some concepts are necessarily overlapping or general. If multiple sibling terms are applicable (*i.e.* terms under a common parent), the parent term may be applicable.
- **3. SHOULD NOT** use both a term and it's parent or other ancestor, when annotating a single attribute. An exception would be a tool which *e.g.* performs some general Sequence analysis operations but specialises on Protein feature detection.

Tip: If you're struggling to find the terms you need, or the meaning of a term is not obvious, search EDAM using the browsers below (they have different functionalities). Multiple searches using synonyms, alternative spellings *etc.* can help.

- EBI OLS browser
- NCBO BioPortal browser
- EDAM Tool Annotator Demo

If you cannot find the right term, request it's added to EDAM via GitHub but first read the guidelines on how to request a term.

Note: It currently takes some time from requesting new EDAM terms for these to be supported in bio.tools. In future, you'll be able to request terms directly via the bio.tools registration interface and these terms will become immediately available for use, albeit subject to approval and possible change before inclusion in EDAM and bio.tools.

Note: Some high-level "organisational" concepts defined in EDAM are intended primarily to structure the hierarchy, and are not intended for annotation in bio.tools. They are defined in EDAM.owl via ''<usageGuideline>Not recommended for annotation in bio.tools.</usageGuideline>'. Such tips are visible in the OLS and BioPortal browsers.

4.2 Attribute guidelines

Guidelines below are organised into sections as they appear in the bio.tools registration user interface

4.2.1 Summary group

Basic information about the software.

Name

Canonical software name assigned by the software developer or service provider, e.g. "needle"

- **1. MUST** use name in common use, *e.g.* in the tool homepage and publication.
- **2. MUST** use short form if available *e.g.* `ExPASy` **not** `ExPASy Bioinformatics Resource Portal`.
- **3. MUST NOT** include general or technical terms ("software", "application", "server", "service", "SOAP", "REST", "RESTful" *etc.*) *unless* these are part of the common name
- **4. MUST NOT** misappropriate the names of other tools, *e.g.* there are many online BLAST services besides the original NCBI BLAST tool; calling any of them "BLAST" would be wrong
- **5. MUST NOT** include version information *unless* this is part of common name
- **6. SHOULD** preserve capitalisation *e.g.* `ExPASy` **not** `expasy`.
- **7. SHOULD** follow the naming patterns (see below)

Note:

- see the syntax guidelines.

Note: Naming patterns

For database portals use the pattern:

`name (acronym)` *e.g.* `The Protein Databank (PDB)`

- a common abbreviation can be given instead of an acronym
- if no common acronym or abbreviation exists, omit this part: do not invent one!

For tools that simply wrap or provide an interface to some other tool, including Web APIs (REST), Web services (SOAP+WSDL), and web applications over command-line tools, use the pattern:

`{collectionName} toolName {API|WS}{( providerName)}` *e.g.* `EMBOSS water API (ebi)`

where:

- `collectionName` is the name of suite, workbench or other collection the underlying tool is from (if applicable)
- `toolName` is the canonical name of the underlying tool
- use `API` for Web APIs or `WS` for Web services
- `providerName` is the name of the institute providing the online service (if applicable)

If in exceptional cases (*i.e.* when registering, as separate entries, versions of a tool with fundamental differences), substitute for `toolName` in the pattern above:

`toolname versionID` *e.g.* `FindPeaks 3.1`

where `versionID` is the version number.

Tip:

- in case of mulitple related entries be consistent, *e.g.* `Open PHACTS` and `Open PHACTS API`
- be wary of names that are very long (>25 characters). If shortening the name is necessary, don't truncate it in a way (*e.g.* within the middle of a word) that would render it meaningless or unintuitive

Short description

Short and concise textual description of the software function, e.g. "Needleman-Wunsch global alignment of two sequences."

- **1. MUST** provide a terse statement of the primary purpose / function of the tool: what is done not how
- **2. MUST** begin with a capital letter and end with a period ('.')
- **3. MUST NOT** include tool name
- **4. MUST NOT** include any of the following, *unless* essential to distinguish the tool from other bio.tool entries:
  - general or technical terms ("software", "application", "server", "service", "SOAP", "REST", "RESTful" *etc.*)
  - provenance information *e.g.* software provider, institute or person name
- **5. MUST NOT** describe how good the software is (mentions of applicability are OK)
- **6. MUST NOT** include URLs
- **7. SHOULD** use declarative sentences (ideally a single sentence!) in the present tense

Note:

- see the syntax guidelines.

Description

Textual description of the software, e.g. "needle reads two input sequences and writes their optimal global sequence alignment to file. It uses the Needleman-Wunsch alignment algorithm to find the optimum alignment (including gaps) of two sequences along their entire length. The algorithm uses a dynamic programming method to ensure the alignment is optimum, by exploring all possible alignments and choosing the best."

- **1. MUST** provide a concise summary of purpose / function of the tool
- **2. MUST** begin with a capital letter and end with a period ('.')
- **4. SHOULD NOT** include any of the following, *unless* essential to distinguish the tool from other bio.tool entries:
  - general or technical terms ("software", "application", "server", "service", "SOAP", "REST", "RESTful" *etc.*)
  - provenance information *e.g.* software provider, institute or person name
- **5. SHOULD NOT** describe how good the software is (mentions of applicability are OK)
- **6. SHOULD NOT** include URLs

Note:

- see the syntax guidelines.

Homepage

Homepage of the software, or some URL that best serves this purpose, e.g. "http://emboss.open-bio.org/rel/rel6/apps/needle.html"

- **1. MUST** resolve to a web page from the developer / provider that most specifically describes the tool

---

Note:

- see the syntax guidelines.

Tip: In case a tool lacks it's own website, a URL of it's code repository is OK. Do not use a general URL such as an institutional homepage, unless nothing better is available.

Version

Version information (typically a version number) of the software applicable to this bio.tools entry, e.g. "6.4.0.0"

- **1. MUST** correctly identify the tool version as described by the other attributes (see note below)
- **2. MUST** specify exactly the public version label in common use
- **3. MUST NOT** include tokens such as "v", "ver", "version", "rel", "release" *etc.*, *unless* these are part of the public version label
- **4. MAY** identify all tool versions which are applicable to the entry

- **5. MAY** specify a version for database portals and web applications, but only if this is used in the common name

---

Note:

- see the syntax guidelines.

Important:

Care is needed to ensure annotations correspond to the indicated tool version.

- **only** change the version if you're sure there's no fundamental change to the specified tool functions (operations, inputs and outputs)
- if there are fundamental changes, update the tool function annotation
- **do not** assume version "1" in case the version number is not readily findable

---

Tip: One or more version fields may be specified, and each - in princple - allows flexible specification of version information including single versions, ranges, lists and lists including ranges, *e.g.*:

- 1.1
- beta01
- 2.0 - 2.7
- 1.1, 1.2.1, 1.4, v5
- 1.1 - 1.4, 2.0-alpha, 2.0-beta-01 - 2.0-beta-04, 2.0.0
- *etc.*

We recommend to keep things simple (one version label per field by default) and pragmatic (using version ranges where desirable).

Other IDs

A unique identifier of the software, typically assigned by an ID-assignment authority other than bio.tools, e.g. "RRID:SCR_015644"

- **1. MUST** correctly identify the same tool as indicated by the biotoolsID
- **2. MUST** include version information if IDs for multiple different versions are specified
- **3. MAY** specify the type of identifier (see below)

Maturity	Description
doi	Digital Object Identifier of the software assigned (typically) by the software developer or service provider.
rrid	Research Resource Identifier as used by the NIH-supported Resource Identification Portal (https://scicrunch.org/resources).
cpe	Common Platform Enumeration (CPE) identifier as listed in the CPE dictionary (https://cpe.mitre.org/dictionary/).
biotoolsCURIE	bio.tools CURIE (secondary identifier).

Note:

- see the syntax guidelines.

Value

Value of tool identifier, e.g. "RRID:SCR_001156"

- **1. MUST** specify a valid identifier for the tool.

Type

Type of tool identifier, e.g. "rrid"

- **1. MAY** specify the applicable type, in terms from a controlled vocabulary (see below) - although this should not normally be necessary

Version

Version information (typically a version number) of the software applicable to this identifier, e.g. "1.4"

- **1. MUST** correctly identify the applicable tool version
- **2. MUST** follow the general guidelines for version

4.2.2 Function group

Details of a function (i.e. mode of operation) the software provides, expressed in terms from the EDAM ontology.

Operation

The basic operation(s) performed by this software function (EDAM Operation), e.g. "'Protein signal peptide detection' (http://edamontology.org/operation_0418)"

- **1. MUST** correctly specify operations performed by the tool, or (if version is indicated), those specific version(s) of the tool
- **2. MUST** be correctly organised into multiple functions, in case the tool has multiple modes of operation (see guidelines for tool functions).
- **3. SHOULD** describe all the primary operations performed by that tool and **SHOULD NOT** describe secondary / minor operations: if in any doubt, mail registry-support.

Attention:

- see the general guidelines for EDAM annotations.

Note:

- see the syntax guidelines.

Data type (input and output data)

Type of primary input / output data (if any) e.g. "'Sequence' (http://edamontology.org/data_2044)"

- **1. MUST** correctly specify types of input or output data processed by the tool, or (if version is indicated), those specific version(s) of the tool
- **2. MUST** be correctly associated with the operation(s); for each function in case the tool has multiple modes of operation (see guidelines for tool functions).
- **3. SHOULD** describe all the primary inputs and outputs of the tool and **SHOULD NOT** describe secondary / minor inputs and outputs: if in any doubt, mail registry-support.

Attention:

- see the general guidelines for EDAM annotations.

Tip:

- many tools allow a primary input to be specified in a number of alternative ways, the common case being a sequence input that may be specified via a sequence identifier, or by typing in a literal sequence. In such cases, annotate the input using the EDAM Data concept for the type of data, not the identifier.

Note:

- see the syntax guidelines for input and output

Data format (input and output data)

Allowed format(s) of primary inputs/outputs e.g. "'FASTA' (http://edamontology.org/format_1929)"

- **1. MUST** correctly specify data formats supported on input or output by the tool, or (if version) is indicated, those specific version(s) of the tool
- **2. MUST** be correctly associated with the data type of an input or output (see guidelines for tool functions).
- **3. SHOULD** describe the primary data formats and **MAY** exhaustively describe *all* formats: if in any doubt, mail registry-support.

Attention: see the general guidelines for EDAM annotations.

Note:

- see the syntax guidelines.

Comment

Concise comment about this function, if not apparent from the software description and EDAM annotations, e.g. "This option is slower, but more precise."

- **1. MUST** not duplicate what is already apparent from the EDAM annotations
- **2. SHOULD** be concise and summarise only critical usage information
- **3. SHOULD NOT** duplicate online documentation; give a link if necessary

---

Note:

- see the syntax guidelines.

Command

Relevant command, command-line fragment or option for executing this function / running the tool in this mode, e.g "-s best"

- **1. MUST** specify precisely a command, command-line fragment or option specified in the tool documentation
- **2. MUST** be correctly associated with a function (the command must be used to invoke that specific tool function)

Note:

- see the syntax guidelines.

4.2.3 Labels group

Miscellaneous scientific, technical and administrative details of the software, expressed in terms from controlled vocabularies.

Tool type

The type of application software: a discrete software entity can have more than one type, e.g. "Command-line tool, Web application"

- **1. MUST** specify all types that are applicable, in terms from a controlled vocabulary (see below)

Type	Description
Command-line tool	A tool with a text-based (command-line) interface.
Database portal	A Web application, suite or workbench providing a portal to a biological database.
Desktop application	A tool with a graphical user interface that runs on your desktop environment, *e.g.* on a PC or mobile device.
Library	A collection of components that are used to construct other tools. bio.tools scope includes component libraries performing high-level bioinformatics functions but excludes lower-level programming libraries.
Ontology	A collection of information about concepts, including terms, synonyms, descriptions etc.
Plug-in	A software component encapsulating a set of related functions, which are not standalone, *i.e.* depend upon other software for its use, *e.g.* a Javascript widget, or a plug-in, extension add-on etc. that extends the function of some existing tool.
Script	A tool written for some run-time environment (*e.g.* other applications or an OS shell) that automates the execution of tasks. Often a small program written in a general-purpose languages (*e.g.* Perl, Python) or some domain-specific languages (*e.g.* sed).
SPARQL endpoint	A service that provides queries over an RDF knowledge base via the SPARQL query language and protocol, and returns results via HTTP.
Suite	A collection of tools which are bundled together into a convenient toolkit. Such tools typically share related functionality, a common user interface and can exchange data conveniently. This includes collections of stand-alone command-line tools, or Web applications within a common portal.
Web application	A tool with a graphical user interface that runs in your Web browser.
Web API	An application programming interface (API) consisting of endpoints to a request-response message system accessible via HTTP. Includes everything from simple data-access URLs to RESTful APIs.
Web service	An API described in a machine readable form (typically WSDL) providing programmatic access via SOAP over HTTP.
Workbench	An application or suite with a graphical user interface, providing an integrated environment for data analysis which includes or may be extended with any number of functions or tools. Includes workflow systems, platforms, frameworks etc.
Workflow	A set of tools which have been composed together into a pipeline of some sort. Such tools are (typically) standalone, but are composed for convenience, for instance for batch execution via some workflow engine or script.

Tip:

- in cases where a given software is described by more than one entry (*e.g.* a web application and its API are described separately) then assign only the types that are applicable to that entry.

Note:

- bio.tools includes all types of bioinformatics tools: application software with well-defined data processing functions (inputs, outputs and operations). When registering a tool, one or more tool types may be assigned, reflecting the different facets of the software being described.
- see the syntax guidelines.

Topic

General scientific domain the software serves or other general category (EDAM Topic), e.g. "'Protein sites, features and motifs' (http://edamontology.org/topic_3510)"

- **1. MUST** specify the single most important and relevant scientific topic
- **2. MAY** specify all highly relevant scientific topics
- **3. SHOULD NOT** exhaustively specify all the topics of lower or secondary relevance

Attention:

- see the general guidelines for EDAM annotations.

Note:

- see the syntax guidelines.

Operating system

The operating system supported by a downloadable software package, e.g. "Linux"

- **1. MUST** specify all operating systems that are applicable, in terms from a controlled vocabulary (see below)

Maturity	Description
Linux	All flavours of Linux/UNIX operating systems.
Windows	All flavours of Microsoft Windows operating system.
Mac	All flavours of Apple Macintosh operating systems (primarily Mac OS X).

Note:

- see the syntax guidelines.

Language

Name of programming language the software source code was written in, e.g. "C"

- **1. MUST** specify the primary language used, in terms from a controlled vocabulary (see below)
- **2. MAY** exhaustively specify other languages used

Note:

- a controlled vocabulary of valid terms is defined in biotoolsSchema.
- see the syntax guidelines.

License

Software or data usage license, e.g. "GPL-3.0"

- **1. MUST** acurately describe the license used.
- **2. SHOULD** use "Proprietary" in cases where the software is under license (not defined in biotoolsSchema) whereby it can be obtained from the provider (*e.g.* for money), and then owned, *i.e.* definitely not an open-source or free software license.
- **3. SHOULD** use "Unlicensed" for software which is not licensed and is not "Proprietary".
- **4. SHOULD** use "Other" if the software is available under a license not listed by biotoolsSchema and which is not "Proprietary".
  - a controlled vocabulary of valid terms is defined in biotoolsSchema.
  - see the syntax guidelines.

Tip: Use the "Other" license for custom institutional licenses that are out of scope of biotoolsSchema. If you've found a license that you think should be included in biotoolsSchema please report it *via* GitHub.

Note: Most permisible values are identifiers from the SPDX license list (https://spdx.org/licenses/). In future, based on the specified license a label (e.g. "Open-source") may be attached to the bio.tools entry (see table below)

Table 4.1: Labelling based on license (future work)

License	Description
Open-source	Software is made available under a license approved by the Open Source Initiative (OSI). The software is distributed in a way that satisfies the 10 criteria of the Open Source Definition maintained by OSI (see https://opensource.org/docs/osd). The source code is available to others.
Free software	Free as in 'freedom' not necessarily free of charge. Software is made available under a license approved by the Free Software Foundation (FSF). The software satisfies the criteria of the Free Software Definition maintained by FSF (see http://www.gnu.org/philosophy/free-sw.html). The source code is available to others.
Free and open source	Software is made available under a license approved by both the Open Source Initiative (OSI) and the Free Software Foundation (FSF), and satisfies the criteria of the OSI Open Source Definition maintained (https://opensource.org/docs/osd) and the FSF Free Software Definition (http://www.gnu.org/philosophy/free-sw.html). Such software ensures users have the freedom to run, copy, distribute, study, change and improve the software. The source code is available to others.
Copyleft	Software is made available under a license designated as 'copyleft' by the Free Software Foundation (FSF). The license ensures such software is free and that all modified and extended versions of the program are free as well. Free as in 'freedom' not necessarily free of charge, as per the Free Software Definition maintained by FSF (see http://www.gnu.org/philosophy/free-sw.html).

Collection

Unique ID of a collection that the software has been assigned to within bio.tools, e.g. "CBS

- **1. SHOUD** be short and intuitive

Tip:

- collections may be created for for any arbitrary purpose
- biotoolsSchema allows tool relationships to be defined, but these are not yet supported in bio.tools. In the meantime, collections may be used to group together related entries.

Note:

- see the syntax guidelines.

Maturity

How mature the software product is, e.g. "Mature"

- **1. MUST** acurately reflect the software maturity, in terms from a controlled vocabulary (see below)

Maturity	Description
Emerging	Nascent or early release software that may not yet be fully featured or stable.
Mature	Software that is generally considered to fulfill several of the following: secure, reliable, actively maintained, fully featured, proven in production environments, has an active community, and is described or cited in the scientific literature.
Legacy	Software which is no longer in common use, deprecated by the provider, superseded by other software, replaced by a newer version, is obsolete etc.

Attention:

- normally only the developer or provider of a tool is sure of its maturity. If you are not sure, then do not complete this field.

Note:

- see the syntax guidelines.

Cost

Monetary cost of acquiring the software, e.g. "Free of charge (with retritions)"

- **1. MUST** acurately describe the monetary cost of acquiring the software, in terms from a controlled vocabulary (see below)

Cost	Description
Free of charge	Software which available for use by all, with full functionality, at no monetary cost to the user.
Free of charge (with restrictions)	Software which is available for use at no monetary cost to the user, but possibly with limited functionality, usage restrictions, or other limitations.
Commercial	Software which you have to pay to access.

Note:

- see the syntax guidelines.

Accessibility

Whether the software is freely available for use, e.g. "Open access"

- **1. MUST** acurately describe the accessibility conditions that apply, in terms from a controlled vocabulary (see below)

Accessibility	Description
Open access	An online service which is available for use to all, but possibly requiring user accounts / authentication.
Restricted access	An online service which is available for use to a restricted audience, e.g. members of a specific institute.
Proprietary	Software for which the software's publisher or another person retains intellectual property rights usually copyright of the source code, but sometimes patent rights.
Freeware	Proprietary software that is available for use at no monetary cost. In other words, freeware may be used without payment but may usually not be modified, re-distributed or reverse-engineered without the author's permission.

Note:

- see the syntax guidelines.

4.2.4 Links group

Miscellaneous links for the software e.g. repository, issue tracker or mailing list.

Note:

- the bio.tools registration interace & API allows a curator to record when a link of a certain type is known to *not* be available
- see the syntax guidelines.

URL

A link of some relevance to the software (URL), e.g. "https://github.com/pharmbio/sciluigi/issues"

- **1. MUST** resolve to a page of the indicated link type
- **2. MUST NOT** give a general link (*e.g.* homepage URL) if a more specific link is available
- **3. MUST NOT** specify a link of a certain type is "Not available" *unless* certain (*i.e.* following a reasonably thorough search) that this is indeed the case

Link type

The type of data, information or system that is obtained when the link is resolved, e.g. "Mailing list"

- **1. MUST** acurately specify the type of information available at the link, in terms from a controlled vocabulary (see below)

Link type	Description
Browser	A website for browsing data.
Helpdesk	Helpdesk providing support in using the software.
Issue tracker	Tracker for software issues, bug reports, feature requests etc.
Mailing list	Mailing list for the software announcements, discussions, support etc.
Mirror	Mirror of an (identical) online service.
Registry	Some registry, catalogue etc. other than bio.tools.
Repository	Repository where source code, data and other files may be downloaded.
Social media	A website used by the software community including social networking sites, discussion and support fora, WIKIs etc.
Scientific benchmark	Information about the scientific performance of a tool.
Technical monitoring	Information about the technical status of a tool.

Comment

Comment about the link, e.g. "Please use the issue tracker for reporting bugs and making features requests."

- **1. SHOULD** be a concise summary of practical information

4.2.5 Download group

Links to downloads for the software, e.g. source code, virtual machine image or container.

Note:

- the bio.tools registration interace & API allows a curator to record when a documentation link of a certain type is known to *not* be available
- see the syntax guidelines.

URL

Link to download (or repo providing a download) for the software, e.g. "http://bioconductor/packages/release/bioc/src/contrib/VanillaICE_1.36.0.tar.gz"

- **1. MUST** resolve to a page providing either an immediately download, or links for a download of the indicated link type
- **2. MUST NOT** give a general link (*e.g.* homepage URL) if a more specific link is available
- **3. MUST NOT** specify a download of a certain type is "Not available" *unless* certain (*i.e.* following a reasonably thorough search) that this is indeed the case

Download type

Type of download that is linked to, e.g. "Binaries"

- **1. MUST** acurately specify the type of download available at the link, in terms from a controlled vocabulary (see below)

Download type	Description
API specification	File providing an API specification for the software, e.g. Swagger/OpenAPI, WSDL or RAML file.
Biological data	Biological data, or a web page on a database portal where such data may be downloaded.
Binaries	Binaries for the software.
Binary package	Binary package for the software.
Command-line specification	File providing a command line specification for the software.
Container file	Container file including the software.
CWL file	Common Workflow Language (CWL) file for the software.
Icon	Icon of the software.
Ontology	A file containing an ontology, controlled vocabulary, terminology etc.
Screenshot	Screenshot of the software.
Source code	Software source code.
Source package	Source package (of various types) for the software.
Test data	Data for testing the software is working correctly.
Test script	Script used for testing testing whether the software is working correctly.
Tool wrapper (galaxy)	Galaxy tool configuration file (wrapper) for the software.
Tool wrapper (taverna)	Taverna configuration file for the software.
Tool wrapper (other)	Workbench configuration file (other than taverna, galaxy or CWL wrapper) for the software.
VM image	Virtual machine (VM) image for the software.

Comment

Comment about the download, e.g. "Complete distibution"

- **1. SHOULD** be concise and summarise only practical information about the link

Cmd

A useful command pertinent to the download, e.g. for getting or installing a tool, e.g. "-s best".

- **1. MUST** be a functional commmand of practical value

Version

Version information (typically a version number) of the software applicable to this download.

- **1. MUST** correctly identify the applicable tool version
- **2. MUST** follow the general guidelines for version

4.2.6 Documentation group

Links to documentation about the software e.g. manual, API specification or training material.

Note:

- the bio.tools registration interace & API allows a curator to record when a documentation link of a certain type is known to *not* be available
- see the syntax guidelines.

URL

Link to documentation on the web for the tool, e.g. "http://bioconductor.org/packages/release/bioc/html/VanillaICE.html"

- **1. MUST** resolve to a page of the indicated documentation type
- **2. MUST NOT** give a general link (*e.g.* homepage URL) if a more specific link is available
- **3. MUST NOT** specify documentation of a certain type is "Not available" *unless* certain (*i.e.* following a reasonably thorough search) that this is indeed the case

Documentation type

Type of documentation that is linked to, e.g. "Citation instructions"

- **1. MUST** acurately specify the type of documentation available at the link, in terms from a controlled vocabulary (see below)

Documentation type	Description
API documentation	Human-readable API documentation.
Citation instructions	Information on how to correctly cite use of the software.
Contributions policy	Information about policy for making contributions to the software project.
General	General documentation.
Governance	Information about the software governance model.
Installation instructions	Instructions how to install the software.
Manual	Information on how to use the software.
Terms of use	Rules that one must agree to abide by in order to use a service.
Training material	Online training material such as text on a Web page, a presentation, video, tutorial etc.
Tutorial	A tutorial about using the software.
Other	Some other type of documentation not listed in biotoolsSchema.

Comment

Comment about the documentation, e.g. "Comprehensive usage information suitable for biologist end-users."

- **1. SHOULD** be concise and summarise only practical information about the link

4.2.7 Publications group

Publications about the software

- **1. MUST** correctly identify a relevant publication
- **2. MUST** specify multiple IDs for a single publication within a single publication group

---

Note:

- see the syntax guidelines.

PubMed Central ID

PubMed Central Identifier (PMCID) of a publication about the software, e.g. "PMC4343077"

PubMed ID

PubMed Identifier (PMID) of a publication about the software, e.g. "21959131"

Digital Object ID

Digital Object Identifier (DOI) of a publication about the software, e.g. "10.1038/nmeth.1701"

Publication type

Type of publication, e.g. "Primary"

- **1. MUST** acurately specify the type of publication, in terms from a controlled vocabulary (see below)

Download type	Description
Primary	The principal publication about the software itself; the article to cite when acknowledging use of the software.
Benchmark	A publication which assessed the performance of the software.
Review	A publication where the software was reviewed.
Other	A publication about the software but not the primary publication or a benchmark study.

Version

Version information (typically a version number) of the software applicable to this publication.

- **1. MUST** correctly identify the applicable tool version
- **2. MUST** follow the general guidelines for version

4.2.8 Credits group

Individuals or organisations that should be credited, or may be contacted about the software.

- **1. SHOULD** provide contact details for the first port-of-call when seeking help with the software

- **2. MAY** specify one or more other credits

Note:

- a credit consists either simply the name of an ELIXIR Platform or ELIXIR node *or* the name of some other entity that is credited, with associated metadata
- see the syntax guidelines.

Name

Name of the entity that is credited, e.g. "EMBL EBI"

- **1. MUST** give the first and last names of a person, or the correct name of some other entity.
- **2. MUST NOT** give a redirect, *e.g.* "See publication", a URL, or any information other than the name of the entity that is credited.

ELIXIR Platform

Name of the ELIXIR Platform that is credited, e.g. "Tools"

- **1. MUST** only credit the ELIXIR Platform if directly contributing to the work, using a term from a controlled vocabulary (see below)

ELIXIR Platform	Description
Data	ELIXIR Data Platform
Tools	ELIXIR Tools Platform
Compute	ELIXIR Compute Platform
Interoperability	ELIXIR Interoperability Platform
Training	ELIXIR Training Platform

ELIXIR Node

Name of the ELIXIR Node that is credited, e.g. "Norway"

- **1. MUST** only credit the ELIXIR Node if directly contributing to the work, using a term from a controlled vocabulary (see below)

ELIXIR Node
Belgium
Czech Republic
Denmark
EMBL
Estonia
Finland
France
Germany
Greece
Hungary
Ireland
Israel
Italy
Luxembourg
Netherlands
Norway
Portugal
Slovenia
Spain
Sweden
Switzerland
UK

ORCID ID

Unique identifier (ORCID iD) of a person that is credited, e.g. "http://orcid.org/0000-0002-1825-0097"

- **1. MUST** correctly identify a credited person

Note: Open Researcher and Contributor IDs (ORCID IDs) provide a persistent reference to information on a researcher, see http://orcid.org/.

GRID ID

Unique identifier (GRID ID) of an organisation that is credited, e.g. "grid.5170.3"

- **1. MUST** correctly identify a credited organisation

Note: Global Research Identifier Database (GRID) IDs provide a persistent reference to information on research organisations, see https://www.grid.ac/. If ORCID institutional identifiers become available, these will also be supported.

Email

Email address of the entity that is credited e.g. "hnielsen@cbs.dtu.dk"

- **1. MUST** specify a syntactically valid email address

- **2. MUST NOT** specify an email address that is not publicly acknowledged as credit for the software, *e.g.* on a webpage or in a publication
- **3. MUST NOT** specify a stale (obsolete) email address

URL

URL for the entity that is credited, e.g. homepage of an institute, e.g. "http://www.ebi.ac.uk/"

- **1. MUST** resolve to a page of information directly relevant to the credited entity

Telephone number

Telephone number of the entity that is credited, e.g. "+49-89-636-48018"

- **1. MUST** specify a valid telephone number
- **2. MUST NOT** specify a telephone number that is not publicly advertised as a contact point for the software, *e.g.* on a webpage or in a publication
- **3. MUST NOT** specify a stale (obsolete) telephone number

Entity type

Type of entity that is credited, e.g. "Person"

- **1. MUST** acurately specify the type of entity that is credited, in terms from a controlled vocabulary (see below)

Entity type	Description
Person	Credit of an individual.
Project	Credit of a community software project not formally associated with any single institute.
Division	Credit of or a formal part of an institutional organisation, e.g. a department, research group, team, etc
Institute	Credit of an organisation such as a university, hospital, research institute, service center, unit etc.
Consortium	Credit of an association of two or more institutes or other legal entities which have joined forces for some common purpose. Includes Research Infrastructures (RIs) such as ELIXIR, parts of an RI such as an ELIXIR node etc.
Funding agency	Credit of a legal entity providing funding for development of the software or provision of an online service.

Role

Role performed by entity that is credited, e.g. "Developer"

- **1. MUST** acurately specify the primary role of credited entity, in terms from a controlled vocabulary (see below)
- **2. MAY** exhaustively specify all the roles of the credited entity

Role	Description
Developer	Author of the original software source code.
Maintainer	Maintainer of a mature software providing packaging, patching, distribution etc.
Provider	Institutional provider of an online service.
Documentor	Author of software documentation including making edits to a bio.tools entry.
Contributor	Some other role in software production or service delivery including design, deployment, system administration, evaluation, testing, documentation, training, user support etc.
Support	Provider of support in using the software.

Comment

A comment about the credit, e.g. "Wrote the user manual."

- **1. SHOULD** be concise and acurate, elaborating on the contribution of the credited entity
- **2. MUST NOT** duplicate information that is, or can, be provided via the `role` attribute, *i.e.* do not specify only "Developer", "Support" *etc.*

4.3 Tool type guidelines

4.3.1 Command-line tool

A tool with a text-based (command-line) interface.

4.3.2 Database portal

A Web application, suite or workbench providing a portal to a biological database.

4.3.3 Desktop application

A tool with a graphical user interface that runs on your desktop environment, e.g. on a PC or mobile device.

4.3.4 Library

A collection of components that are used to construct other tools. bio.tools scope includes component libraries performing high-level bioinformatics functions but excludes lower-level programming libraries.

4.3.5 Ontology

A collection of information about concepts, including terms, synonyms, descriptions etc.

- pick one or more topics that best describe the scientific areas covered by the ontology
- pick the operation of "Query and retrieval" (http://edamontology.org/operation_0224)
- do not annotate the type or format of the input and output data

4.3.6 Plug-in

A software component encapsulating a set of related functions, which are not standalone, *i.e.* depend upon other software for its use, e.g. a Javascript widget, or a plug-in, extension add-on etc. that extends the function of some existing tool.

Note:

- biotoolsSchema allows tool relationships to be defined, but these are not yet supported in bio.tools. In future, the `isPluginFor` relationship will allow specification of the tool to which the plug-in is applicable.

4.3.7 Script

A tool written for some run-time environment (e.g. other applications or an OS shell) that automates the execution of tasks. Often a small program written in a general-purpose languages (e.g. Perl, Python) or some domain-specific languages (e.g. sed).

4.3.8 SPARQL endpoint

A service that provides queries over an RDF knowledge base via the SPARQL query language and protocol, and returns results via HTTP.

- pick one or more topics that best describe the underyling data
- pick the operation of "Query and retrieval" (http://edamontology.org/operation_0224)
- do not annotate the type or format of the input and output data

---

Note:

- biotoolsSchema allows tool relationships to be defined, but these are not yet supported in bio.tools. In future, the `isInterfaceTo` relationship will allow specification of the data resource (database portal) that a SPARQL endpoint provides an interface to.

4.3.9 Suite

A collection of tools which are bundled together into a convenient toolkit. Such tools typically share related functionality, a common user interface and can exchange data conveniently. This includes collections of stand-alone command-line tools, or Web applications within a common portal.

- describe the attributes that are commmon to the suite as a whole, not (typically) attributes of individual tools
- individual tools included in the suite should be registered as separate entries
- when annotating the operation of the suite, select operations that are core function of the suite itself / common to all tools in the suite. Alternatively pick one or two of the primary operation(s) of the included tools
- entries for the suite itself and it's component tools can be associated by annotatong them as part of a common collection

Tip: If you are considering to register a suite with many tools, it is a good idea to discuss this first with the bio.tools admin.

Note:

- biotoolsSchema allows tool relationships to be defined, but these are not yet supported in bio.tools. In future, the `includes` relationship will allow specification of the tools that are included in a suite.

Attention: **do not** annotate the type and **'format <>'_** of input and output data, *unless* all tools in the suite happen to have these in common

4.3.10 Web application

A tool with a graphical user interface that runs in your Web browser.

Note:

- biotoolsSchema allows tool relationships to be defined, but these are not yet supported in bio.tools. In future, the `isInterfaceTo` and `uses` relationships will allow specification of the tools that a web application provides an interface to or uses.
- for software that essentially just wraps or provides an interface to some other tool, *e.g.* a web application or web service over an existing tool, use the pattern `toolName providerName` where `providerName` is a name (without spaces) of some institute, workbench, collection *etc.*, *e.g.* `cufflinks cloudIFB`. **Do not** misappropriate the original name!

4.3.11 Web API

An application programming interface (API) consisting of endpoints to a request-response message system accessible via HTTP. Includes everything from simple data-access URLs to RESTful APIs.

- in general, describe the attributes of the API as a whole, not individual endpoint of the API (see note below)
- in case the API has a single endpoint only, the input(s), operation(s) and output(s) may be annotated
- in case the API has many endpoints, annotate the primary operation(s), but **not** the inputs and outputs
- annotate the location of machine-readable API specification (*e.g.* openAPI file) using the download attribute with download type of `API specification` - annotate the location of any human-readable documentation using the documentation attribute with documentation type of `API specification`
- when assigning the name, use the pattern `name API` *e.g.* `Open PHACTS API`
- in case the web service provides an interface to an existing tool registered in bio.tools, try to ensure the relevant annotations are consistent

Note:

- biotoolsSchema includes a basic model of an API specification including endpoints however this is not yet supported in bio.tools

- biotoolsSchema allows tool relationships to be defined, but these are not yet supported in bio.tools. In future, the `isInterfaceTo` relationship will allow specification of the tool or data resource (database portal) that the web service provides an interface to.

4.3.12 Web service

An API described in a machine readable form (typically WSDL) providing programmatic access via SOAP over HTTP.

- in general, describe the attributes of the web service as a whole, not individual endpoint of the service (see note below)
- in case the web service has a single endpoint only, the input(s), operation(s) and output(s) may be annotated
- in case the web service has many endpoints, annotate the primary operation(s), but **not** the inputs and outputs
- annotate the location of the WSDL file using the download attribute with download type of `API specification`
- annotate the location of any human-readable documentation using the documentation attribute with documentation type of `API specification`
- when assigning the name, use the pattern `name WS` *e.g.* `EMMA WS`
- in case the web service provides an interface to an existing tool registered in bio.tools, try to ensure the relevant annotations are consistent

Note:

- biotoolsSchema includes a basic model of an API specification including endpoints however this is not yet supported in bio.tools
- biotoolsSchema allows tool relationships to be defined, but these are not yet supported in bio.tools. In future, the `isInterfaceTo` relationship will allow specification of the tool that the web service provides an interface to

4.3.13 Workbench

An application or suite with a graphical user interface, providing an integrated environment for data analysis which includes or may be extended with any number of functions or tools. Includes workflow systems, platforms, frameworks etc.

- describe the attributes of the workbench as a whole, not (typically) individual tools or functions provided by it
- individual tools included in the workbench, especially where these tools are indepepdently available, should be registered as separate entries
- individual functions provided by the workbench, especially where these are not independently available, should each be described in their own function
- entries for the workbench itself and it's component tools can be associated by annotatong them as part of a common collection

Tip: If you are considering to register a complicated workbench with many tools or functions, it is a good idea to discuss this first with the bio.tools admin.

Note:

- biotoolsSchema allows tool relationships to be defined, but these are not yet supported in bio.tools. In future, the `includes` relationship will allow specification of the tools that are included in a workbench.

4.3.14 Workflow

A set of tools which have been composed together into a pipeline of some sort. Such tools are (typically) standalone, but are composed for convenience, for instance for batch execution via some workflow engine or script.

- when deciding how to annotate a workflow inputs, operations and outputs, consider the worfklow as a "black box" , *i.e.* annotate the input(s) to, output(s) from and primary operation(s) of the workflow as a whole

Note:

- bio.tools does not currently contain many examples of workflows. We welcome input on how to describe worfklows and ensure good coverage: please get in touch with us.
- biotoolsSchema allows tool relationships to be defined, but these are not yet supported in bio.tools. In future, the `includes` relationship will allow specification of the tools that are included in a workflow.

Important: workflows can contain many tools; **do not** list all the operations performed by these tools, just the main operation(s) of the workflow as a whole.

4.4 Further guidelines (bio.tools admin only)

Attention: The guidelines that follow are for attributes and other aspects under the control of bio.tools admin. If you're not a bio.tools admin you can ignore this section.

4.4.1 biotoolsID

Unique ID (case insensitive) of the tool that is assigned upon registration of the software in bio.tools, normally identical to tool name, e.g. "needle".

Attention:

- the ID by default is a URL-safe version of the tool name, and is set (and can only be changed) by bio.tools admin.

- **MUST** use the default value where possible
- **MUST** be clean and intuitive (in case use of default is not possible)
- **MUST NOT** truncate the name (in the middle of a word, or at all) if this renders the ID ugly or meaningless

Note: Transformation rules

The following rules apply when transforming the supplied tool name:

- replace ' ' (spaces) in the name with underscores (a single underscore in case of multiple spaces)
- preserve all reserved characters (uppercase and lowercase letters, decimal digits, hyphen, period, underscore, and tilde), but remove other characters
- use '_' to delimit parts of names but only *if* these are not already truncated in the original name
- adhere to the same patterns for tool name, *e.g.* `EMBOSS_water_API_ebi`

4.4.2 biotoolsCURIE

bio.tools CURIE (compact URI) based on the unique bio.tools ID of the tool, e.g. "biotools:needle"

Note:

- identical to biotoolsID but with the prefix `biotools:`

CHAPTER 5

Editors Guide

Thematic editors are registry-core members responsible for overseeing coverage and quality in specific thematic areas. The editorships will enable expanding accurate high-standard software annotations in bio.tools to most scientific topics in life science.

5.1 Background

The sheer number of software and continuous advancements in tool development demand extensive and sustained efforts to provide comprehensive annotations. Such challenges can be overcome by distributing the curation effort across a network of engaged and experienced partners, which contribute to improve bio.tools content and to adjust the EDAM vocabulary (used in bio.tools) to the needs of the respective communities. Careful coordination of the distributed tasks is required to keep similar quality standards and increase tool interoperability.

Thematic editors are well connected with their respective scientific community, experts in their field, have a broad knowledge about commonly used software and are motivated to promote bio.tools. The following tasks will enforce an improvement to the bio.tools content:

- Review of bio.tools and EDAM to survey conceptual and semantic coverage.
- Development of a strategy to achieve and sustain a minimum information level, as per the emerging information standard (Section 3.4, Appendix C). Updates and documentation have been provided through assigned Sifterapp issues (e.g. RNA analysis tools, proteomics tools, livestock and plant tools and Metabolomics tools).
- Engage with their community, supervise hackathons and promote bio.tools in general.
- Mentor a student for practical work such as manual curation.
- Implement sustainable procedures for systematic tool reviews (e.g. bio.tools alert service) and curation standards.

Benefits: A thematic editorships provides the opportunity to become a documented expert. Editors will acquire extensive knowledge and experience about available software in their fields *e.g.* to publish high-quality reviews. The editors will train additional experts during training and curation events (*e.g.* hackathons) as well as *via* student supervision. Moreover, the ELIXIR framework will support them to apply for further funding to amplify their activities within their field (at least, this will be pursued).

5.2 Current thematic editors

Editor	Topics
Anne Wenzel	RNA structure
Jon Ison	General purpose sequence analysis
Jose M. Carazo	Electron microscopy, structural biology
Josep Gelpi	Structural bioinformatics
Jurgen Haas	Protein structural biology, structural bioinformatics
Marta Villegas	Natural language processing
Martin Krallinger	Text Mining, natural language processing, named entity recognition
Reza Salek	Metabolomics
Veit Schw?mmle	Proteomics, statistics
Vivi R. Gregersen	Agricultural science (association analysis, genomics)

bio.tools studentships can support thematic editors. A typical studentship comprises (at least) one full month of full working hours which are mostly distributed over a longer time period. Applications are written on basis of the template (see *e.g.* proteomics studentship). The students are recruited after proposal dissemination *via* GitHub, project mailing lists *etc.*

CHAPTER 6

Documentors Guide

Documentation for bio.tools and related projects are maintained in GitHub:

https://github.com/bio-tools/biotoolsDocs

Documentation files are written in **reStructuredText** and have the file extension `.rst`. Uploading a file to GitHub will trigger a rebuild of the docs. GitHub include the file `index.rst` which defines the menu structure. Each menu item corresponds to a GitHub file, which (by convention) should have the same name as the menu item: use concise names!

6.1 reStructuredText links

- Quick Reference
- Primer
- Full documentation
- Online editor
- readthedocs FAQ
- readthedocs : getting started
- readthedocs build process
- thread on wide table handling

CHAPTER 7

API endpoints

This is a lightweight web service with a REST interface, which provides an easy way to access the bio.tools database. An API (Application programming interface) is a protocol intended to be used as an interface by software components to communicate with each other.

If you find a bug, have any questions or suggestions, please get in touch with us.

7.1 List resources

List and search through all the available resources. Can sort, filter, and search the results.

HTTP GET

```
https://bio.tools/api/tool/
https://bio.tools/api/t/
```

7.1.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
page	No	Integer	1	Result page number
format	No	String(json, xml, api)	json	Response media type
q	No	String		Query term, used for searching, matches all attributes
sort	No	String(lastUpdate, additionDate, name, affiliation, score)	lastUpdate	Sorts the results by choosen value (score only available when there is a query)
ord	No	String(desc, asc)	desc	Orders the results by either Ascending or Descending order
<attribute>	No	String		Filter by <attribute>. List of supported attributes below.

7.1.2 Filtering

To filter the results by attribute name, the attribute name has to be added as a parameter to the URL, with the value being the desired search term, e.g. `?name=signalp`

Attributes

These are attributes supported by bio.tools

```
id, name, topic, function, operation, input,
inputDataFormat, inputDataType, output, outputDataFormat,
outputDataType, homepage, description, version,
accessibility, toolType, collection, contact,
elixirInfo, maturity, operatingSystem, language,
cost, license, documentation, link, download, publication,
credit, owner
```

7.1.3 Example

```
curl -X GET "https://bio.tools/api/tool/?page=1&format=json&name=signalp&sort=name&
↪ord=asc&q=protein-signal-peptide-detection"
```

7.1.4 Response data

Key Name	Description	Example
count	The total resource count results for your query	2313
previous	Link to the previous page	?page=4
next	Link to the next page	?page=6
list	An array with multiple resources and their relative information	ARRAY

7.2 Resource detail

Obtain information about a single resource.

HTTP GET

```
https://bio.tools/api/tool/:id/
https://bio.tools/api/t/:id/
```

7.2.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
id	Yes	String		Resource unique ID
format	No	String(json, xml, api)	json	Response media type

7.2.2 Example

```
curl -X GET "https://bio.tools/api/tool/signalp/?format=json"
```

7.3 List resource versions

Obtain information about avaliable versions of a single resource.

HTTP GET

```
https://bio.tools/api/tool/:id/version/
https://bio.tools/api/t/:id/version/
```

7.3.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
id	Yes	String		Resource unique ID
format	No	String(json, xml, api)	json	Response media type

7.3.2 Example

```
curl -X GET "https://bio.tools/api/t/signalp/version/"
```

7.4 Resource version detail

Obtain information about a specified version of a single resource.

HTTP GET

```
https://bio.tools/api/tool/:id/version/:version_id
https://bio.tools/api/t/:id/version/:version_id
```

7.4.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
id	Yes	String		Resource unique ID
format	No	String(json, xml, api)	json	Response media type
version_id	Yes	String		Resource version unique ID

7.4.2 Example

```
curl -X GET "https://bio.tools/api/tool/signalp/version/4.1?format=json"
```

7.5 Register a resource

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP POST

```
https://bio.tools/api/tool/
https://bio.tools/api/t/
```

7.5.1 Endpoint Parameters

Parameter	Required	Type	Description
data	Yes	Resource	Resource you wish to register. See an example resource.

Note: It is possible to specify editing permissions for resources. Learn how to manage *Editing permissions*.

Note: It is possible to create multiple versions of the same resource. Learn how to use *Resource versioning*.

7.5.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	Resource media type
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

7.5.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
-d '<resource>' "https://bio.tools/api/tool/"
```

7.6 Validate registering a resource

Test registering a resource without it actually being saved into the database.

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP POST

```
https://bio.tools/api/tool/validate/
https://bio.tools/api/t/validate/
```

7.6.1 Endpoint Parameters

Parameter	Required	Type	Description
data	Yes	Resource	Resource you wish to validate. See an example resource.

7.6.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	Resource media type
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

7.6.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
-d '<resource>' "https://bio.tools/api/tool/validate/"
```

7.7 Update a resource

Update a resource description.

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP PUT

```
https://bio.tools/api/tool/:id/
https://bio.tools/api/t/:id/
```

7.7.1 Endpoint Parameters

Parameter	Required	Type	Description
id	Yes	String	Resource unique ID
data	Yes	Resource	Description with which you wish to update the resource See an example resource.

Note: It is possible to specify editing permissions for resources. Learn how to manage *Editing permissions*.

Note: It is possible to create multiple versions of the same resource. Learn how to use *Resource versioning*.

7.7.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	Resource media type
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

7.7.3 Example

```
curl -X PUT -H "Content-Type: application/json" \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
-d '<resource>' "https://bio.tools/api/tool/SignalP"
```

7.8 Validate updating a resource

Test updating a resource without it actually being saved into the database.

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP PUT

```
https://bio.tools/api/tool/:id/validate/
https://bio.tools/api/t/:id/validate/
```

7.8.1 Endpoint Parameters

Parameter	Required	Type	Description
id	Yes	String	Resource unique ID
data	Yes	Resource	Description with which you wish to update the resource for validation See an example resource.

7.8.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	Resource media type
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

7.8.3 Example

```
curl -X PUT -H "Content-Type: application/json" \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
-d '<resource>' "https://bio.tools/api/tool/SignalP/validate/"
```

7.9 Resource versioning

All resources can have a specified version assigned to them. This allows for example new versions of resources to be registered while keeping an older version of the resource intact. In order to create a new version of a given resource, the following parameters need to be added to the resource data:

Parameter	Type	Description
versionId	String	ID of a new resource version to be created. Once created, the version id becomes permanent and is used to uniquely identify a specific version. See *Resource version detail* for more information.
latest	1 or 0	Specify if the created resource version is the latest. All previous resources marked as 'latest' will no longer be considered that after a new version gets marked as 'latest'.

7.10 Editing permissions

It is possible to manage editing permissions for the registered resources. There are currently three types of editing permissions supported by the system:

7.10.1 Private

A private resource can only be edited by the creator of the resource. This is the default option. In order to set this kind of permission, add the following info into the resource data:

```
"editPermission": {
    "type": "private"
}
```

7.10.2 Public

Public resource can be modified by any user registered in the system. In order to set this kind of permission, add the following info into the resource data:

```
"editPermission": {
    "type": "public"
}
```

7.10.3 Group

Specify a list of users in the system that can edit the resource. In order to set this kind of permission, add the following info into the resource data:

```
"editPermission": {
    "type": "private",
    "authors": [
        "registered_user_1", "registered_user_2"
    ]
}
```

7.11 Delete a resource

Removes a resource from the registry.

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP DELETE

```
https://bio.tools/api/tool/:id/
https://bio.tools/api/t/:id/
```

7.11.1 Endpoint Parameters

Parameter	Required	Type	Description
id	Yes	String	Resource unique ID

7.11.2 Headers

Parameter	Required	Allowed values	Description
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

7.11.3 Example

```
curl -X DELETE \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
"https://bio.tools/api/tool/SignalP"
```

7.12 List used terms

Obtain a list of terms registered with tools for some attributes, e.g. a list of names of all tools.

HTTP GET

```
https://bio.tools/api/used-terms/:attribute
```

7.12.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
attribute	Yes	String(name, topic, functionName, input, output, credits, all)		Attribute for which a list of used terms will be returned
format	No	String(json, xml, api)	json	Response media type

7.12.2 Example

```
curl -X GET "https://bio.tools/api/used-terms/name/?format=json"
```

7.12.3 Response data

Key Name	Description
data	A list of used terms

7.13 Create a user account

Creates a user account and emails a verification email.

HTTP POST

```
https://bio.tools/api/rest-auth/registration/
```

7.13.1 POST data

Key Name	Required	Type	Description	Example
username	Yes	String	Account username	username
password1	Yes	String	Password	password
password2	Yes	String	Repeated password	password
email	Yes	String	Account email. The verification email will be sent to this address	example@example.org

7.13.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

7.13.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"username":"username", "password1":"password", \
"password2":"password", "email":"example@example.org"}' \
"https://bio.tools/api/rest-auth/registration/"
```

7.14 Verify a user account

Verifies a user account based on the emailed verification key.

HTTP POST

```
https://bio.tools/api/rest-auth/registration/verify-email/
```

7.14.1 POST data

Key Name	Required	Type	Description	Example
key	Yes	String	Verification key from account creation email	ndwowtbpmlk5zxdxfrwgu2822xynjidhizhwosycve7hro1of156hjwdsf1f6

7.14.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

7.14.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"key":"ndwowtbpmlk5zxdxfrwgu2822xynjidhizhwosycve7hro1of156hjwdsf1f6gbn"}' \
"https://bio.tools/api/rest-auth/registration/verify-email/"
```

7.15 Log in / obtain token

Logs the user in and returns an authentication token.

HTTP POST

```
https://bio.tools/api/rest-auth/login/
```

7.15.1 POST data

Key Name	Required	Type	Description	Example
username	Yes	String	Account username	username
password	Yes	String	Password	password

7.15.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

7.15.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"username":"username","password":"password"}' \
"https://bio.tools/api/rest-auth/login/"
```

7.15.4 Response data

Key Name	Description
key	Authentication token

7.16 Get user information

Returns information about the logged in user account, including a list of registered resource (name, id, version, additionDate, lastUpdate)

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP GET

```
https://bio.tools/api/rest-auth/user/
```

7.16.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
format	No	String(json, xml, api)	json	Response media type

7.16.2 Headers

Parameter	Required	Allowed values	Description
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

7.16.3 Example

```
curl -X GET \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
"https://bio.tools/api/rest-auth/user/?format=json"
```

7.16.4 Response data

Key Name	Description
username	Account username
email	Account email
resources	List of registered resources (limited to name, id, version, additionDate, lastUpdate)

7.17 Log out

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP POST

```
https://bio.tools/api/rest-auth/logout/
```

7.17.1 Headers

Parameter	Required	Allowed values	Description
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

7.17.2 Example

```
curl -X POST
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
"https://bio.tools/api/rest-auth/logout/"
```

7.18 Reset user password

Sends a password reset email.

HTTP POST

```
https://bio.tools/api/rest-auth/password/reset/
```

7.18.1 POST data

Key Name	Required	Type	Description	Example
email	Yes	String	Account email	example@example.org

7.18.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

7.18.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"email":"example@example.org"}' \
"https://bio.tools/api/rest-auth/password/reset/"
```

7.19 Confirm password reset

Confirms a password reset using uid and token from a password reset email.

HTTP POST

```
https://bio.tools/api/rest-auth/password/reset/confirm/
```

7.19.1 POST data

Key Name	Required	Type	Description	Example
uid	Yes	String	UID from password reset email	MQ
token	Yes	String	Token from password reset email	4ct-67e90a1ab4f22fbb9b9f
password1	Yes	String	New password	new_password
password2	Yes	String	New password repeated	new_password

7.19.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

7.19.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"uid":"MQ", "token":"4ct-67e90a1ab4f22fbb9b9f", \
"password1":"new_password", "password2":"new_password"}' \
"https://bio.tools/api/rest-auth/password/reset/confirm/"
```

7.20 Stats

Compile stats about a the registry.

HTTP GET

```
https://bio.tools/api/stats
```

7.20.1 Example

```
curl -X GET "https://bio.tools/api/stats/?format=json"
```

CHAPTER 8

API endpoints - development

Note: This is the API documentation for upcoming features, available on the dev server at https://dev.bio.tools.

This is a lightweight web service with a REST interface, which provides an easy way to access the bio.tools database. An API (Application programming interface) is a protocol intended to be used as an interface by software components to communicate with each other.

If you find a bug, have any questions or suggestions, please get in touch with us.

8.1 List resources

List and search through all the available resources. Can sort, filter, and search the results.

HTTP GET

```
https://dev.bio.tools/api/tool/
https://dev.bio.tools/api/t/
```

8.1.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
page	No	Integer	1	Result page number
format	No	String(json, xml, api)	json	Response media type
q	No	String		Query term, used for searching, matches all attributes
sort	No	String(lastUpdate, additionDate, name, affiliation, score)	lastUpdate	Sorts the results by choosen value (score only available when there is a query)
ord	No	String(desc, asc)	desc	Orders the results by either Ascending or Descending order
<attribute>	No	String		Filter by <attribute>. List of supported attributes below.

8.1.2 Filtering

To filter the results by attribute name, the attribute name has to be added as a parameter to the URL, with the value being the desired search term, e.g. `?name=signalp`

Attributes

These are attributes supported by bio.tools

```
id, name, topic, function, operation, input,
inputDataFormat, inputDataType, output, outputDataFormat,
outputDataType, homepage, description, version,
accessibility, toolType, collection, contact,
elixirInfo, maturity, operatingSystem, language,
cost, license, documentation, link, download, publication,
credit, owner
```

8.1.3 Example

```
curl -X GET "https://dev.bio.tools/api/tool/?page=1&format=json&name=signalp&
↪sort=name&ord=asc&q=protein-signal-peptide-detection"
```

8.1.4 Response data

Key Name	Description	Example
count	The total resource count results for your query	2313
previous	Link to the previous page	?page=4
next	Link to the next page	?page=6
list	An array with multiple resources and their relative information	ARRAY

8.2 Resource detail

Obtain information about a single resource.

HTTP GET

```
https://dev.bio.tools/api/tool/:id/
https://dev.bio.tools/api/t/:id/
https://dev.bio.tools/api/:id/
```

8.2.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
id	Yes	String		Resource unique ID
format	No	String(json, xml, api)	json	Response media type

8.2.2 Example

```
curl -X GET "https://dev.bio.tools/api/tool/signalp/?format=json"
```

8.3 Register a resource

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP POST

```
https://dev.bio.tools/api/tool/
https://dev.bio.tools/api/t/
```

8.3.1 Endpoint Parameters

Parameter	Required	Type	Description
data	Yes	Resource	Resource you wish to register. See an example resource.

Note: It is possible to specify editing permissions for resources. Learn how to manage *Editing permissions*.

8.3.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	Resource media type
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

8.3.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
-d '<resource>' "https://dev.bio.tools/api/tool/"
```

8.4 Validate registering a resource

Test registering a resource without it actually being saved into the database.

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP POST

```
https://dev.bio.tools/api/tool/validate/
https://dev.bio.tools/api/t/validate/
```

8.4.1 Endpoint Parameters

Parameter	Required	Type	Description
data	Yes	Resource	Resource you wish to validate. See an example resource.

8.4.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	Resource media type
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

8.4.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
-d '<resource>' "https://dev.bio.tools/api/tool/validate/"
```

8.5 Update a resource

Update a resource description.

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP PUT

```
https://dev.bio.tools/api/tool/:id/
https://dev.bio.tools/api/t/:id/
https://dev.bio.tools/api/:id/
```

8.5.1 Endpoint Parameters

Parameter	Required	Type	Description
id	Yes	String	Resource unique ID
data	Yes	Resource	Description with which you wish to update the resource See an example resource.

Note: It is possible to specify editing permissions for resources. Learn how to manage *Editing permissions*.

8.5.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	Resource media type
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

8.5.3 Example

```
curl -X PUT -H "Content-Type: application/json" \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
-d '<resource>' "https://dev.bio.tools/api/tool/SignalP"
```

8.6 Validate updating a resource

Test updating a resource without it actually being saved into the database.

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP PUT

```
https://dev.bio.tools/api/tool/:id/validate/
https://dev.bio.tools/api/t/:id/validate/
https://dev.bio.tools/api/:id/validate/
```

8.6.1 Endpoint Parameters

Parameter	Required	Type	Description
id	Yes	String	Resource unique ID
data	Yes	Resource	Description with which you wish to update the resource for validation See an example resource.

8.6.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	Resource media type
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

8.6.3 Example

```
curl -X PUT -H "Content-Type: application/json" \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
-d '<resource>' "https://dev.bio.tools/api/tool/SignalP/validate/"
```

8.7 Editing permissions

It is possible to manage editing permissions for the registered resources. There are currently three types of editing permissions supported by the system:

8.7.1 Private

A private resource can only be edited by the creator of the resource. This is the default option. In order to set this kind of permission, add the following info into the resource data:

```
"editPermission": {
    "type": "private"
}
```

8.7.2 Public

Public resource can be modified by any user registered in the system. In order to set this kind of permission, add the following info into the resource data:

```
"editPermission": {
    "type": "public"
}
```

8.7.3 Group

Specify a list of users in the system that can edit the resource. In order to set this kind of permission, add the following info into the resource data:

```
"editPermission": {
    "type": "private",
    "authors": [
        "registered_user_1", "registered_user_2"
    ]
}
```

8.8 Delete a resource

Removes a resource from the registry.

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP DELETE

```
https://dev.bio.tools/api/tool/:id/
https://dev.bio.tools/api/t/:id/
https://dev.bio.tools/api/:id/
```

8.8.1 Endpoint Parameters

Parameter	Required	Type	Description
id	Yes	String	Resource unique ID

8.8.2 Headers

Parameter	Required	Allowed values	Description
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

8.8.3 Example

```
curl -X DELETE \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
"https://dev.bio.tools/api/tool/SignalP"
```

8.9 List used terms

Obtain a list of terms registered with tools for some attributes, e.g. a list of names of all tools.

HTTP GET

```
https://dev.bio.tools/api/used-terms/:attribute
```

8.9.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
attribute	Yes	String(name, topic, functionName, input, output, credits, all)		Attribute for which a list of used terms will be returned
format	No	String(json, xml, api)	json	Response media type

8.9.2 Example

```
curl -X GET "https://dev.bio.tools/api/used-terms/name/?format=json"
```

8.9.3 Response data

Key Name	Description
data	A list of used terms

8.10 Create a user account

Creates a user account and emails a verification email.

HTTP POST

```
https://dev.bio.tools/api/rest-auth/registration/
```

8.10.1 POST data

Key Name	Required	Type	Description	Example
username	Yes	String	Account username	username
password1	Yes	String	Password	password
password2	Yes	String	Repeated password	password
email	Yes	String	Account email. The verification email will be sent to this address	example@example.org

8.10.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

8.10.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"username":"username", "password1":"password", \
"password2":"password", "email":"example@example.org"}' \
"https://dev.bio.tools/api/rest-auth/registration/"
```

8.11 Verify a user account

Verifies a user account based on the emailed verification key.

HTTP POST

```
https://dev.bio.tools/api/rest-auth/registration/verify-email/
```

8.11.1 POST data

Key Name	Required	Type	Description	Example
key	Yes	String	Verification key from account creation email	ndwowtbpmlk5zxdxfrwgu2822xynjidhizhwosycve7hro1of156hjwdsf1f6

8.11.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

8.11.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"key":"ndwowtbpmlk5zxdxfrwgu2822xynjidhizhwosycve7hro1of156hjwdsf1f6gbn"}' \
"https://dev.bio.tools/api/rest-auth/registration/verify-email/"
```

8.12 Log in / obtain token

Logs the user in and returns an authentication token.

HTTP POST

```
https://dev.bio.tools/api/rest-auth/login/
```

8.12.1 POST data

Key Name	Required	Type	Description	Example
username	Yes	String	Account username	username
password	Yes	String	Password	password

8.12.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

8.12.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"username":"username","password":"password"}' \
"https://dev.bio.tools/api/rest-auth/login/"
```

8.12.4 Response data

Key Name	Description
key	Authentication token

8.13 Get user information

Returns information about the logged in user account, including a list of registered resource (name, id, version, additionDate, lastUpdate)

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP GET

```
https://dev.bio.tools/api/rest-auth/user/
```

8.13.1 Endpoint Parameters

Parameter	Required	Type	Default	Description
format	No	String(json, xml, api)	json	Response media type

8.13.2 Headers

Parameter	Required	Allowed values	Description
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

8.13.3 Example

```
curl -X GET \
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
"https://dev.bio.tools/api/rest-auth/user/?format=json"
```

8.13.4 Response data

Key Name	Description
username	Account username
email	Account email
resources	List of registered resources (limited to name, id, version, additionDate, lastUpdate)

8.14 Log out

Note: This method requires the user to be authenticated. Learn how to *Log in / obtain token*.

HTTP POST

```
https://dev.bio.tools/api/rest-auth/logout/
```

8.14.1 Headers

Parameter	Required	Allowed values	Description
Authorization	Yes	String('Token <authorization token>')	Authorization header. Learn how to *Log in / obtain token*.

8.14.2 Example

```
curl -X POST
-H "Authorization: Token 028595d682541e7e1a5dcf2306eccb720dadafd7" \
"https://dev.bio.tools/api/rest-auth/logout/"
```

8.15 Reset user password

Sends a password reset email.

HTTP POST

```
https://dev.bio.tools/api/rest-auth/password/reset/
```

8.15.1 POST data

Key Name	Required	Type	Description	Example
email	Yes	String	Account email	example@example.org

8.15.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

8.15.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"email":"example@example.org"}' \
"https://dev.bio.tools/api/rest-auth/password/reset/"
```

8.16 Confirm password reset

Confirms a password reset using uid and token from a password reset email.

HTTP POST

```
https://dev.bio.tools/api/rest-auth/password/reset/confirm/
```

8.16.1 POST data

Key Name	Required	Type	Description	Example
uid	Yes	String	UID from password reset email	MQ
token	Yes	String	Token from password reset email	4ct-67e90a1ab4f22fbb9b9f
password1	Yes	String	New password	new_password
password2	Yes	String	New password repeated	new_password

8.16.2 Headers

Parameter	Required	Allowed values	Description
Content-Type	Yes	String(application/json, application/xml)	POST data media type

8.16.3 Example

```
curl -X POST -H "Content-Type: application/json" \
-d '{"uid":"MQ", "token":"4ct-67e90a1ab4f22fbb9b9f", \
"password1":"new_password", "password2":"new_password"}' \
"https://dev.bio.tools/api/rest-auth/password/reset/confirm/"
```

8.17 Stats

Compile stats about a the registry.

HTTP GET

```
https://dev.bio.tools/api/stats
```

8.17.1 Example

```
curl -X GET "https://dev.bio.tools/api/stats"
```

CHAPTER 9

Attribute model - development

Attention: UNDER CONSTRUCTION

- guidelines for the bio.tools API
- or to make suggestions about these guidelines please add comments via GitHub

Note: This is the API documentation for upcoming features, coming soon to https://bio.tools.

This page documents the structure of a XML / JSON / YAML file that describes a tool for submission to https://bio.tools.

9.1 Payload formats

To submit a tool via our bio.tools API you'll need to POST a tool description to the tool endpoint. The API supports XML, JSON and YAML format uploads comatible with biotoolsSchema (https://github.com/bio-tools/biotoolsschema).

9.1.1 XML

A sample XML document may look like this:

9.1.2 JSON

A sample JSON document may look like this:

 # THIS EXAMPLE IS WRONG … WILL FIX SOON


```
{ "name": "SignalP", "topic": [
        { "uri": "http://edamontology.org/topic_0003", "term": "Topic"
        }
    ], "function": [
        { "comment": "predicts the presence and location of signal peptide cleavage sites in amino acid
          sequences from different organisms", "handle": "–someOption", "operation": [
            { "uri": "http://edamontology.org/operation_0418", "term": "Protein signal peptide de-
            tection"
            }, {
              "uri": "http://edamontology.org/operation_0422", "term": "Protein cleavage site
              prediction"
            }
          ], "input": [
            { "data": {
                "uri": "http://edamontology.org/data_2044", "term": "Sequence"
              }, "format": [
                { "uri": "http://edamontology.org/format_1929", "term": "FASTA"
                }
              ]
            }
          ], "output": [
              { "data": {
                  "uri": "http://edamontology.org/data_1277", "term": "Protein features"
                }, "format": [
                  { "uri": "http://edamontology.org/format_2305", "term": "GFF"
                  }
                ]
              }, {
                "data": {
                  "uri": "http://edamontology.org/data_2955", "term": "Sequence re-
                  port"
                }, "format": [
                  { "uri": "http://edamontology.org/format_2305", "term": "GFF"
                  }
                ]
```

```
            }
        ]
    }
], "homepage": "http://cbs.dtu.dk/services/SignalP/", "description": "Prediction of the presence and location of signal peptide cleavage sites in amino acid sequences from different organisms.", "cost": "Free of charge (with restrictions)", "maturity": "Mature", "credit": [
    { "name": "TN Petersen", "email": "test@email.com", "orcidid": "test", "gridid": "test", "typeEntity": "Person", "typeRole": "Developer", "comment": "test"
    }
], "link": [
    { "url": "http://www.cbs.dtu.dk/cgi-bin/sw_request?signalp", "type": "Repository", "comment": "test"
    }
], "download": [
    { "url": "http://www.cbs.dtu.dk/cgi-bin/sw_request?signalp", "type": "Source code", "comment": "test"
    }, {
        "url": "http://www.cbs.dtu.dk/cgi-bin/sw_request?signalp", "type": "Binaries", "comment": "test"
    }
], "license": "Proprietary", "operatingSystem": [
    "Linux", "Mac"
], "toolType": [
    "Command-line tool", "Web application"
], "language": [
    "ActionScript"
], "documentation": [
    { "url": "http://www.cbs.dtu.dk/services/SignalP", "type": "General", "comment": "test"
    }
], "publication": [
    { "pmcid": "21959131", "pmid": "21959131", "doi": "doi:10.1038/nmeth.1701", "type": "Primary"
    }, {
        "pmcid": "21959131", "pmid": "21959131", "doi": "doi:10.1038/nmeth.1701", "type": "Other"
    }
], "collectionID": [
    "CBS"
], "contact": [
```

{ “email”: “hnielsen@cbs.dtu.dk”, “name”: “Henrik Nielsen”, “tel”: “123456798”, “url”: “https://bio.tools”

}

], “editPermission”: {

“type”: “private”, “authors”: [“ekry”]

}

}

9.1.3 YAML

A sample YAML document may look like this:

9.2 Tool attributes

9.2.1 Name

Canonical software name assigned by the software developer or service provider, e.g. “needle”

Attribute name name

Required Yes

Type String

Restrictions Min length: 1

Max length: 100

Pattern: [p{Zs}A-Za-z0-9+.,-_:;()]*

Example

```
# XML
<name>needle</name>

# JSON
"name": "needle"
```

Note:

- name may only contain space, uppercase and lowercase letters, decimal digits, plus symbol, period, comma, dash, underscore, colon, semicolon and parentheses.
- line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines.

9.2.2 Short description

Short and concise textual description of the software function, e.g. "Needleman-Wunsch global alignment of two sequences."

Attribute name shortDescription

Required No

Type String

Restrictions Min length: 10

Max length: 100

Example

```
# XML
<shortDescription>Needleman-Wunsch global alignment of two sequences.</
↪shortDescription>

# JSON
"shortDescription": "Needleman-Wunsch global alignment of two sequences."
```

Note:

- minimum 10 and maximum 100 characters.
- line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines.

9.2.3 Description

Textual description of the software, e.g. "needle reads two input sequences and writes their optimal global sequence alignment to file. It uses the Needleman-Wunsch alignment algorithm to find the optimum alignment (including gaps) of two sequences along their entire length. The algorithm uses a dynamic programming method to ensure the alignment is optimum, by exploring all possible alignments and choosing the best."

Attribute name description

Required Yes

Type String

Restrictions Min length: 10

Max length: 500

Example

```
# XML
<shortDescription>needle reads two input sequences and writes their optimal global␣
↪sequence alignment to file. It uses the Needleman-Wunsch alignment algorithm to␣
↪find the optimum alignment (including gaps) of two sequences along their entire␣
↪length. The algorithm uses a dynamic programming method to ensure the alignment is␣
↪optimum, by exploring all possible alignments and choosing the best.</
↪shortDescription>
```

```
# JSON
"shortDescription": "needle reads two input sequences and writes their optimal global␣
↪sequence alignment to file. It uses the Needleman-Wunsch alignment algorithm to␣
↪find the optimum alignment (including gaps) of two sequences along their entire␣
↪length. The algorithm uses a dynamic programming method to ensure the alignment is␣
↪optimum, by exploring all possible alignments and choosing the best."
```

Note:

- minimum 10 and maximum 500 characters.
- line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines.

9.2.4 Homepage

Homepage of the software, or some URL that best serves this purpose, e.g. "http://emboss.open-bio.org/rel/rel6/apps/needle.html"

Attribute name homepage

Required Yes

Type URL

Restrictions Pattern: http(s?)://[^s/$.?#].[^s]*

Example

```
# XML
<homepage>http://emboss.open-bio.org/rel/rel6/apps/needle.html</homepage>

# JSON
"homepage": "http://emboss.open-bio.org/rel/rel6/apps/needle.html"
```

Note:

- a single valid URL is specified.
- see the curation guidelines.

9.2.5 biotoolsID

Unique ID (case insensitive) of the tool that is assigned upon registration of the software in bio.tools, normally identical to tool name, e.g. "needle".

Attribute name biotoolsID

Required No

Type String

Restrictions Pattern: [_-.0-9a-zA-Z]*

Example

```
# XML
<biotoolsID>needle</biotoolsID>

# JSON
"biotoolsID": "needle"
```

Attention:

- a biotoolsID is set (and can only be changed) by bio.tools admin. It can be retrieved by API, but if specified in the payload to a `PUT` or `POST` request will be disregarded.

Note:

- the biotoolssID is a URL-safe and Linked-Data-safe derivative of (often identical to) the tool name. Allowed characters are uppercase and lowercase English letters (case insensitive!), decimal digits, hyphen, period, and underscore. Spaces can be preserved as underscore ("_").
- see the curation guidelines.

9.2.6 biotoolsCURIE

bio.tools CURIE (compact URI) based on the unique bio.tools ID of the tool, e.g. "biotools:needle"

Attribute name biotoolsCURIE

Required No

Type String

Restrictions Pattern: biotools:[_-.0-9a-zA-Z]*

Example

```
# XML
<biotoolsCURIE>needle</biotoolsCURIE>

# JSON
"biotoolsCURIE": "needle"
```

Attention:

- a biotoolsCURIE is set (and can only be changed) by bio.tools admin. It can be retrieved by API, but if specified in the payload to a `PUT` or `POST` request will be disregarded.

Note:

- the bio.tools CURIE is simply the bio.tools tool ID with the prefix "biotools:".
- see the curation guidelines.

9.2.7 Version

Version information (typically a version number) of the software applicable to this bio.tools entry, e.g. "6.4.0.0"

Attribute name version

Required No

Type String

Restrictions Min length: 1

Max length: 100

Pattern: [p{Zs}A-Za-z0-9+.,-_:;()]*

Example

```
# XML
<version>6.4.0.0</version>
<version>1.1 - 1.4, 2.0-alpha, 2.0-beta-01 - 2.0-beta-04, 2.0.0</version>

# JSON
"version":
[
  "6.4.0.0",
  "1.1 - 1.4, 2.0-alpha, 2.0-beta-01 - 2.0-beta-04, 2.0.0"
]
```

Note:

- name may only contain space, uppercase and lowercase English letters, decimal digits, plus symbol, period, comma, dash, colon, semicolon and parentheses.
- line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines.

9.2.8 Other IDs

A unique identifier of the software, typically assigned by an ID-assignment authority other than bio.tools, e.g. "RRID:SCR_015644"

Attribute name otherID

Required No

Type List of otherID objects

otherID object definition

- **value**
  - Required: Yes
  - Type: String
- **type**
  - Required: No

  - Type: ENUM (list)
  - Allowed values (see Curators Guide) - `doi` - `rrid` - `cpe` - `biotoolsCURIE`
- **version**
  - Required: No
  - Type: String
  - Restrictions: Min length: 1, Max length: 100
  - Pattern: [p{Zs}A-Za-z0-9+.,-_:;()]*

Example

```
# XML
<otherID>
      <value>RRID:SCR_015644</value>
      <type>rrid</type>
      <version>4.1</version>
</otherID>
<otherID>
      <value>10.1007/978-1-4939-7015-5_6</value>
      <type>doi</type>
      <version>4.1</version>
</otherID>

# JSON
"otherID":
[
      {
          "value": "RRID:SCR_015644",
          "type:" "rrid",
          "version": "4.1"
      },
      {
          "value": "10.1007/978-1-4939-7015-5_6",
          "type": "doi"
          "version": "4.1"
      }
]
```

Note:

- type can normally be inferred from the value but should be specified otherwise. In the example it was not actually necessary to specify "type".
- see the curation guidelines.

9.2.9 Function

Details of a function (i.e. mode of operation) the software provides, expressed in terms from the EDAM ontology.

Attribute name function

Required No

Type List of function objects

Function object definition

Content

- *Operation*
  - Required: Yes
  - Type: List of EDAM objects
- *Input*
  - Required: No
  - Type: List of input objects
- *Output*
  - Required: No
  - Type: List of output objects
- **comment**
  - Required: No
  - Type: String
  - Restrictions: min length: 10, max length: 1000
- **cmd**
  - Required: No
  - Type: String
  - Restrictions: min length: 1, max length: 100

Note:

- **comment** and **cmd**: line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines for the function group, comment and command.

Example

```
# XML
<function>
    <operation>
        <uri>http://edamontology.org/operation_0418</uri>
        <term>Protein signal peptide detection</term>
    </operation>
    <operation>
        <uri>http://edamontology.org/operation_0422</uri>
        <term>Protein cleavage site prediction</term>
    </operation>
    <input>
        <data>
          <uri>http://edamontology.org/data_2044</uri>
          <term>Sequence</term>
        </data>
        <format>
            <uri>http://edamontology.org/format_1929</uri>
```

```
            <term>FASTA</term>
        </format>
    <output>
        <data>
          <uri>http://edamontology.org/data_1277</uri>
          <term>Protein features</term>
        </data>
        <format>
            <uri>http://edamontology.org/format_2305</uri>
            <term>GFF</term>
        </format>
        <data>
          <uri>http://edamontology.org/data_2955</uri>
          <term>Sequence report</term>
        </data>
        <format>
            <uri>http://edamontology.org/format_1929</uri>
            <term>FASTA</term>
        </format>
    </output>
    <comment>Predicts the presence and location of signal peptide cleavage sites in␣
↪amino acid sequences from different organisms.</comment>
    <cmd>-s best</cmd>
</function>


# JSON
"function":
[
  {
    "operation":
    [
      {
        "uri": "http://edamontology.org/operation_0418",
        "term": "Protein signal peptide detection"
      },
      {
        "uri": "http://edamontology.org/operation_0422",
        "term": "Protein cleavage site prediction"
      }
    ],
    "input":
    [
      {
        "data":
        {
          "uri": "http://edamontology.org/data_2044",
          "term": "Sequence"
        },
        "format":
        [
          {
            "uri": "http://edamontology.org/format_1929",
            "term": "FASTA"
          }
        ]
      }
    ],
```

```
    "output":
    [
      {
        "data":
        {
          "uri": "http://edamontology.org/data_1277",
          "term": "Protein features"
        },
        "format":
        [
          {
            "uri": "http://edamontology.org/format_2305",
            "term": "GFF"
          }
        ]
      },
      {
        "data":
        {
          "uri": "http://edamontology.org/data_2955",
          "term": "Sequence report"
        },
        "format":
        [
          {
            "uri": "http://edamontology.org/format_1929",
            "term": "FASTA"
          }
        ]
      }
    ]
    "comment": "Predicts the presence and location of signal peptide cleavage sites␣
↪in amino acid sequences from different organisms.",
    "cmd": "-s best",
  }
]
```


Operation

The basic operation(s) performed by this software function (EDAM Operation), e.g. "'Protein signal peptide detection' (http://edamontology.org/operation_0418)"

Attribute name operation

Required Yes

Child of *Function*

Type List of EDAM objects

EDAM object definition

Content

- **uri**
  - Required: No (if term present), Yes (otherwise)
  - Type: URL

- **term**
  - Required: No (if URI present), Yes (otherwise)
  - Type: String

Note:

- an EDAM ontology Operation concept URL and / or term are specified, *e.g.* "Multiple sequence alignment", http://edamontology.org/operation_0492.
- URI and term are validated against EDAM ontology; if term and URI do not match, an error will be returned.
- synonyms of terms (as defined in EDAM) are accepted, however, **the synonym will be replaced with main term**.
- see the curation guidelines.

Example

```
# XML
<operation>
        <uri>http://edamontology.org/operation_0418</uri>
        <term>Protein signal peptide detection</term>
</operation>
<operation>
        <uri>http://edamontology.org/operation_0422</uri>
        <term>Protein cleavage site prediction</term>
</operation>

# JSON
"operation":
[
    {
        "uri": "http://edamontology.org/operation_0418",
        "term": "Protein signal peptide detection"
    },
    {
        "uri": "http://edamontology.org/operation_0422",
        "term": "Protein cleavage site prediction"
    }
]
```

Input

Primary input data (if any)

Attribute name input

Required No

Child of *Function*

Type List of input objects

Input object definition

Content

- **data**

- Required: Yes
- Type: EDAM object

- **format**
  - Required: No
  - Type: List of EDAM objects

Example

```
# XML
    <data>
      <uri>http://edamontology.org/data_2044</uri>
      <term>Sequence</term>
    </data>
    <format>
        <uri>http://edamontology.org/format_1929</uri>
        <term>FASTA</term>
    </format>

# JSON
"input":
[
  {
    "data":
    {
      "uri": "http://edamontology.org/data_2044",
      "term": "Sequence"
    },
    "format":
    [
      {
        "uri": "http://edamontology.org/format_1929",
        "term": "FASTA"
      }
    ]
  }
]
```

Output

Primary output data (if any)

Attribute name output

Required No

Child of *Function*

Type List of output objects

Output object definition

Content

- **data**
  - Required: Yes
  - Type: EDAM object

- **format**
  - Required: No
  - Type: List of EDAM objects

Example

```
# XML
"output":
    <data>
      <uri>http://edamontology.org/data_2044</uri>
      <term>Sequence</term>
    </data>
    <format>
        <uri>http://edamontology.org/format_1929</uri>
        <term>FASTA</term>
    </format>

# JSON
"output":
[
  {
    "data":
    {
      "uri": "http://edamontology.org/data_2044",
      "term": "Sequence"
    },
    "format":
    [
      {
        "uri": "http://edamontology.org/format_1929",
        "term": "FASTA"
      }
    ]
  }
]
```

Data

EDAM Data concept, e.g. "'Sequence' (http://edamontology.org/data_2044)" Attribute name

data

Required Yes

Child of *Input* or *Output*

Type EDAM object

EDAM object definition

Content

- **uri**
  - Required: No (if term present), Yes (otherwise)
  - Type: URL
- **term**

- Required: No (if URI present), Yes (otherwise)
- Type: String

Note:

- an EDAM ontology Data concept URL and / or term are specified, *e.g.* "Protein sequences", http://edamontology.org/data_2976.
- URI and term are validated against EDAM ontology; if term and URI do not match, an error will be returned.
- synonyms of terms (as defined in EDAM) are accepted, however, **the synonym will be replaced with main term**.
- see the curation guidelines.

Example

```
# XML
<data>
 <uri>http://edamontology.org/data_2044</uri>
 <term>Sequence</term>
</data>

# JSON
"data":
{
  "uri": "http://edamontology.org/data_2044",
  "term": "Sequence"
}
```

Format

EDAM Format concept, e.g. "'FASTA' (http://edamontology.org/format_1929)"

Attribute name format

Required No

Child of *Input* or *Output*

Type List of EDAM objects

EDAM object definition

Content

- **uri**
  - Required: No (if term present), Yes (otherwise)
  - Type: URL
- **term**
  - Required: No (if URI present), Yes (otherwise)
  - Type: String

Note:

- an EDAM ontology Format concept URL and / or term are specified, *e.g.* "FASTA", http://edamontology.org/format_1929.
- URI and term are validated against EDAM ontology; if term and URI do not match, an error will be returned.
- synonyms of terms (as defined in EDAM) are accepted, however, **the synonym will be replaced with main term**.
- see the curation guidelines.

Example

```
# XML
<format>
 <uri>http://edamontology.org/format_1929</uri>
 <term>FASTA</term>
</format>

# JSON
"format":
[
  {
    "uri": "http://edamontology.org/format_1929",
    "term": "FASTA"
  }
]
```

9.2.10 Tool type

The type of application software: a discrete software entity can have more than one type, e.g. "Command-line tool, Web application"

Attribute name toolType

Required Yes

Type ENUM (list)

Allowed values (see Curators Guide)

- `Command-line tool`
- `Database portal`
- `Desktop application`
- `Library`
- `Ontology`
- `Plug-in`
- `Script`
- `SPARQL endpoint`
- `Suite`
- `Web application`
- `Web API`
- `Web service`

- `Workbench`
- `Workflow`

Example

```
# XML
<toolType>Command-line tool</toolType>
<toolType>Web application</toolType>

# JSON
"toolType":
[
  "Command-line tool",
  "Web application"
]
```

Note:

- see the curation guidelines.

9.2.11 Topic

General scientific domain the software serves or other general category (EDAM Topic), e.g. "'Protein sites, features and motifs' (http://edamontology.org/topic_3510)"

Attribute name topic

Required No

Type List of EDAM objects

EDAM object definition

Content

- **uri**
  - Required: No (if term present), Yes (otherwise)
  - Type: URL
- **term**
  - Required: No (if URI present), Yes (otherwise)
  - Type: String

Example

```
# XML
<topic>
  <uri>http://edamontology.org/topic_0605</uri>
  <term>Informatics</term>
</topic>
<topic>
  <uri>http://edamontology.org/topic_3303</uri>
  <term>Medicine</term>
</topic>
```

```
# JSON
"topic":
[
  {
    "uri": "http://edamontology.org/topic_0605",
    "term": "Informatics"
  },
  {
    "uri": "http://edamontology.org/topic_3303",
    "term": "Medicine"
  }
]
```

Note:

- an EDAM ontology Topic concept URL and / or term are specified, *e.g.* "Proteomics", http://edamontology.org/topic_0121.
- URI and term are validated against EDAM ontology; if term and URI do not match, an error will be returned.
- synonyms of terms (as defined in EDAM) are accepted, however, **the synonym will be replaced with main term**.
- see the curation guidelines.

9.2.12 Operating system

The operating system supported by a downloadable software package, e.g. "Linux"

Attribute name operatingSystem

Required No

Type ENUM (list)

Allowed values (see Curators Guide)

- `Linux`
- `Windows`
- `Mac`

Example

```
# XML
<operatingSystem>Linux</operatingSystem>
<operatingSystem>Mac</operatingSystem>

# JSON
"operatingSystem":
[
  "Linux",
  "Mac"
]
```

Note:

- see the curation guidelines.

9.2.13 Language

Name of programming language the software source code was written in, e.g. "C"

Attribute name language

Required No

Type ENUM (list)

Allowed values (see Curators Guide) `ActionScript`, `Ada`, `AppleScript`, `Assembly language`, `AWK`, `Bash`, `C`, `C#`, `C++`, `COBOL`, `ColdFusion`, `CWL`, `D`, `Delphi`, `Dylan`, `Eiffel`, `Forth`, `Fortran`, `Groovy`, `Haskell`, `Icarus`, `Java`, `Javascript`, `JSP`, `LabVIEW`, `Lisp`, `Lua`, `Maple`, `Mathematica`, `MATLAB`, `MLXTRAN`, `NMTRAN`, `Pascal`, `Perl`, `PHP`, `Prolog`, `PyMOL`, `Python`, `R`, `Racket`, `REXX`, `Ruby`, `SAS`, `Scala`, `Scheme`, `Shell`, `Smalltalk`, `SQL`, `Turing`, `Verilog`, `VHDL`, `Visual Basic`, `Other`

Example

```
# XML
<language>Python</language>
<language>C</language>

# JSON
"language":
[
  "Python",
  "C"
]
```

Note:

- see the curation guidelines.

9.2.14 License

Software or data usage license, e.g. "GPL-3.0"

Attribute name license

Required No

Type ENUM

Allowed values (see Curators Guide) `0BSD`, `AAL`, `ADSL`, `AFL-1.1`, `AFL-1.2`, `AFL-2.0`, `AFL-2.1`, `AFL-3.0`, `AGPL-1.0`, `AGPL-3.0`, `AMDPLPA`, `AML`, `AMPAS`, `ANTLR-PD`, `APAFML`, `APL-1.0`, `APSL-1.0`, `APSL-1.1`, `APSL-1.2`, `APSL-2.0`, `Abstyles`, `Adobe-2006`, `Adobe-Glyph`, `Afmparse`, `Aladdin`, `Apache-1.0`, `Apache-1.1`, `Apache-2.0`, `Artistic-1.0`, `Artistic-1.0-Perl`, `Artistic-1.0-cl8`, `Artistic-2.0`, `BSD-2-Clause`, `BSD-2-Clause-FreeBSD`, `BSD-2-Clause-NetBSD`, `BSD-3-Clause`, `BSD-3-Clause-Attribution`, `BSD-3-Clause-Clear`, `BSD-3-Clause-LBNL`, `BSD-3-Clause-No-Nuclear-License`, `BSD-3-Clause-No-Nuclear-License-2014`, `BSD-3-Clause-No-Nuclear-Warranty`, `BSD-4-Clause`, `BSD-4-Clause-UC`, `BSD-Protection`, `BSD-Source-Code`, `BSL-1.0`, `Bahyph`, `Barr`, `Beerware`, `BitTorrent-1.0`, `BitTorrent-1.1`, `Borceux`, `CATOSL-1.1`, `CC-BY-1.0`, `CC-BY-2.0`, `CC-BY-2.5`,

CC-BY-3.0, CC-BY-4.0, CC-BY-NC-1.0, CC-BY-NC-2.0, CC-BY-NC-2.5, CC-BY-NC-3.0, CC-BY-NC-4.0, CC-BY-NC-ND-1.0, CC-BY-NC-ND-2.0, CC-BY-NC-ND-2.5, CC-BY-NC-ND-3.0, CC-BY-NC-ND-4.0, CC-BY-NC-SA-1.0, CC-BY-NC-SA-2.0, CC-BY-NC-SA-2.5, CC-BY-NC-SA-3.0, CC-BY-NC-SA-4.0, CC-BY-ND-1.0, CC-BY-ND-2.0, CC-BY-ND-2.5, CC-BY-ND-3.0, CC-BY-ND-4.0, CC-BY-SA-1.0, CC-BY-SA-2.0, CC-BY-SA-2.5, CC-BY-SA-3.0, CC-BY-SA-4.0, CC0-1.0, CDDL-1.0, CDDL-1.1, CECILL-1.0, CECILL-1.1, CECILL-2.0, CECILL-2.1, CECILL-B, CECILL-C, CNRI-Jython, CNRI-Python, CNRI-Python-GPL-Compatible, CPAL-1.0, CPL-1.0, CPOL-1.02, CUA-OPL-1.0, Caldera, ClArtistic, Condor-1.1, Crossword, CrystalStacker, Cube, D-FSL-1.0, DOC, DSDP, Dotseqn, ECL-1.0, ECL-2.0, EFL-1.0, EFL-2.0, EPL-1.0, EUDatagrid, EUPL-1.0, EUPL-1.1, Entessa, ErlPL-1.1, Eurosym, FSFAP, FSFUL, FSFULLR, FTL, Fair, Frameworx-1.0, FreeImage, GFDL-1.1, GFDL-1.2, GFDL-1.3, GL2PS, GPL-1.0, GPL-2.0, GPL-3.0, Giftware, Glide, Glulxe, HPND, HaskellReport, IBM-pibs, IJG, IPA, IPL-1.0, ISC, ImageMagick, Imlib2, Info-ZIP, Intel, Intel-ACPI, Interbase-1.0, JSON, JasPer-2.0, LAL-1.2, LAL-1.3, LGPL-2.0, LGPL-2.1, LGPL-3.0, LGPLLR, LPL-1.0, LPL-1.02, LPPL-1.0, LPPL-1.1, LPPL-1.2, LPPL-1.3a, LPPL-1.3c, Latex2e, Leptonica, LiLiQ-P-1.1, LiLiQ-R-1.1, LiLiQ-Rplus-1.1, Libpng, MIT, MIT, MIT-advertising, MIT-enna, MIT-feh, MITNFA, MPL-1.0, MPL-1.1, MPL-2.0, MPL-2.0-no-copyleft-exception, MS-PL, MS-RL, MTLL, MakeIndex, MirOS, Motosoto, Multics, Mup, NASA-1.3, NBPL-1.0, NCSA, NGPL, NLOD-1.0, NLPL, NOSL, NPL-1.0, NPL-1.1, NPOSL-3.0, NRL, NTP, Naumen, NetCDF, Newsletr, Nokia, Noweb, Nunit, OCCT-PL, OCLC-2.0, ODbL-1.0, OFL-1.0, OFL-1.1, OGTSL, OLDAP-1.1, OLDAP-1.2, OLDAP-1.3, OLDAP-1.4, OLDAP-2.0, OLDAP-2.0.1, OLDAP-2.1, OLDAP-2.2, OLDAP-2.2.1, OLDAP-2.2.2, OLDAP-2.3, OLDAP-2.4, OLDAP-2.5, OLDAP-2.6, OLDAP-2.7, OLDAP-2.8, OML, OPL-1.0, OSET-PL-2.1, OSL-1.0, OSL-1.1, OSL-2.0, OSL-2.1, OSL-3.0, OpenSSL, PDDL-1.0, PHP-3.0, PHP-3.01, Plexus, PostgreSQL, Python-2.0, QPL-1.0, Qhull, RHeCos-1.1, RPL-1.1, RPL-1.5, RPSL-1.0, RSA-MD, RSCPL, Rdisc, Ruby, SAX-PD, SCEA, SGI-B-1.0, SGI-B-1.1, SGI-B-2.0, SISSL, SISSL-1.2, SMLNJ, SMPPL, SNIA, SPL-1.0, SWL, Saxpath, Sendmail, SimPL-2.0, Sleepycat, Spencer-86, Spencer-94, Spencer-99, SugarCRM-1.1.3, TCL, TMate, TORQUE-1.1, TOSL, UPL-1.0, Unicode, Unlicense, VOSTROM, VSL-1.0, Vim, W3C, W3C-19980720, WTFPL, Watcom-1.0, Wsuipa, X11, XFree86-1.1, XSkat, Xerox, Xnet, YPL-1.0, YPL-1.1, ZPL-1.1, ZPL-2.0, ZPL-2.1, Zed, Zend-2.0, Zimbra-1.3, Zimbra-1.4, Zlib, bzip2-1.0.5, bzip2-1.0.6, curl, diffmark, dvipdfm, eGenix, gSOAP-1.3b, gnuplot, iMatix, libtiff, mpich2, psfrag, psutils, xinetd, xpp, zlib-acknowledgement, Proprietary, Other, Unlicensed.

Example

```
# XML
<license>Proprietary</license>

# JSON
"license": "Proprietary"
```

Note:

- see the curation guidelines.

9.2.15 Collection

Unique ID of a collection that the software has been assigned to within bio.tools, e.g. "CBS

Attribute name collectionID

Required No

Type List of strings

Restrictions Min length: 1

Max length: 100

Pattern: [p{Zs}A-Za-z0-9+.,-_:;()]*

Example

```
# XML
<collectionID>CBS</collectionID>
<collectionID>NorduGrid</collectionID>

# JSON
"collectionID":
[
  "CBS",
  "NorduGrid"
]
```

Note:

- collection may only contain space, uppercase and lowercase letters, decimal digits, plus symbol, period, comma, dash, underscore, colon, semicolon and parentheses.
- line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines.

9.2.16 Maturity

How mature the software product is, e.g. "Mature"

Attribute name maturity

Required No

Type ENUM

Allowed valuse (see Curators Guide)

- `Emerging`
- `Mature`
- `Legacy`

Example

```
# XML
<maturity>Mature</maturity>

# JSON
"maturity": "Mature"
```

Note:

- see the curation guidelines.

9.2.17 Cost

Monetary cost of acquiring the software, e.g. "Free of charge (with retritions)"

Attribute name cost

Required No

Type ENUM

Allowed values (see Curators Guide)

- `Free of charge`
- `Free of charge (with restrictions)`
- `Commercial`

Example

```
# XML
<cost>Free of charge (with restrictions)</cost>

# JSON
"cost": "Free of charge (with restrictions)"
```

Note:

- see the curation guidelines.

9.2.18 Accessibility

Whether the software is freely available for use, e.g. "Open access"

Attribute name accessibility

Required No

Type ENUM (list)

Allowed values (see Curators Guide)

- `Open access`
- `Restricted access`
- `Proprietary`
- `Freeware`

Example

```
# XML
<accessibility>Open access</accessibility>
<accessibility>Freeware</accessibility>

# JSON
"accessibility":
```

```
[
  "Open access",
  "Freeware"
]
```

Note:

- see the curation guidelines.

9.2.19 Link

Miscellaneous links for the software e.g. repository, issue tracker or mailing list.

Attribute name link

Required No

Type List of link objects

Link object definition

Content

- **url**
  - Required: Yes
  - Type: URL
- **type**
  - Required: Yes
  - Type: ENUM
  - Allowed values: (see Curators Guide)
    - `Browser`
    - `Helpdesk`
    - `Issue tracker`
    - `Mailing list`
    - `Mirror`
    - `Registry`
    - `Repository`
    - `Social media`
    - `Scientific benchmark`
    - `Technical monitoring`
- **comment**
  - Required: No
  - Type: String
  - Restrictions: min length: 10, max length: 1000

Example

```
# XML
<link>
 <url>http://www.cbs.dtu.dk/cgi-bin/sw_request?signalp</url>
 <type>Repository</type>
 <comment>Source code for current and old versions.</comment>
</link>

# JSON
"link":
[
  {
    "url": "http://www.cbs.dtu.dk/cgi-bin/sw_request?signalp",
    "type": "Repository",
    "comment": "Source code for current and old versions."
  }
]
```

Example

```
# XML
<link>
 <isAvailable>Not available</isAvailable>
 <type>Repository</type>
</download>

# JSON
"link":
[
  {
    "isAvailable": "Not available"
    "type": "Repository"
  }
]
```

Note:

- if a link of a certain type is known to *not* be available, this can be specified using the `isAvailable` attribute (see Example)
- the comment is minimum 10 and maximum 1000 characters. Line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines.

9.2.20 Download

Links to downloads for the software, e.g. source code, virtual machine image or container.

Attribute name download

Required No

Type List of download objects

Download object definition

Content

- **url**
  - Required: Yes
  - Type: URL
- **type**
  - Required: Yes
  - Type: ENUM
  - Allowed values: (see Curators Guide)
    - `API specification`
    - `Biological data`
    - `Binaries`
    - `Binary package`
    - `Command-line specification`
    - `Container file`
    - `CWL file`
    - `Icon`
    - `Ontology`
    - `Screenshot`
    - `Source code`
    - `Source package`
    - `Test data`
    - `Test script`
    - `Tool wrapper (galaxy)`
    - `Tool wrapper (taverna)`
    - `Tool wrapper (other)`
    - `VM image`
- **comment**
  - Required: No
  - Type: String
  - Restrictions: min length: 10, max length: 1000
- **cmd**
  - Required: No
  - Type: String
  - Restrictions: min length: 1, max length: 100
- **version**
  - Required: No

- Type: String
- Restrictions: Min length: 1, Max length: 100
- Pattern: [p{Zs}A-Za-z0-9+.,-_:;()]*

Example

```
# XML
<download>
 <url>http://www.cbs.dtu.dk/cgi-bin/sw_request?signalp</url>
 <type>Source code</url>
 <comment>Complete distibution</comment>
 <cmd>n/a</cmd>
 <version>1.4</version>
</download>

# JSON
"download":
[
  {
    "url": "http://www.cbs.dtu.dk/cgi-bin/sw_request?signalp",
    "type": "Source code",
    "comment": "Complete distibution",
    "cmd": "n/a",
    "version": "1.4"
  }
]
```

Example

```
# XML
<download>
 <isAvailable>Not available</isAvailable>
 <type>Source code</type>
</download>

# JSON
"download":
[
  {
    "isAvailable": "Not available"
    "type": "Source code"
  }
]
```

Note:

- if a download link of a certain type is known to *not* be available, this can be specified using the `isAvailable` attribute (see Example)
- the comment is minimum 10 and maximum 1000 characters. Line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines.

9.2.21 Documentation

Links to documentation about the software e.g. manual, API specification or training material.

Attribute name documentation

Required No

Type List of documentation objects

Documentation object definition

Content

- **url**
  - Required: Yes
  - Type: URL
- **type**
  - Required: Yes
  - Type: ENUM
  - Allowed values: (see Curators Guide)
    - `API documentation`
    - `Citation instructions`
    - `Contributions policy`
    - `General`
    - `Manual`
    - `Terms of use`
    - `Training material`
    - `Other`
- **comment**
  - Required: No
  - Type: String
  - Restrictions: min legth:10, max length: 1000

Example

```
# XML
<documentation>
 <url>http://www.cbs.dtu.dk/services/SignalP</url>
 <type>General</type>
 <comment>Comprehensive usage instructions.</comment>
</documentation>

# JSON
"documentation":
[
  {
    "url": "http://www.cbs.dtu.dk/services/SignalP",
    "type": "General",
```

```
    "comment": "Comprehensive usage instructions"
  }
]
```

Example

```
# XML
<documentation>
 <isAvailable>Not available</isAvailable>
 <type>General</type>
</documentation>

# JSON
"documentation":
[
  {
    "isAvailable": "Not available"
    "type": "General"
  }
]
```

Note:

- if a documentation link of a certain type is known to *not* be available, this can be specified using the `isAvailable` attribute (see Example)
- the comment is minimum 10 and maximum 1000 characters. Line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines.

9.2.22 Publication

Publications about the software

Attribute name publication

Required Yes

Type List of publication objects

Publication object definition

Content

- **pmcid**
  - Required: No
  - Type: PMCID
  - Pattern: (PMC)[1-9][0-9]{0,8}
- **pmid**
  - Required: No
  - Type: PMID
  - Pattern: [1-9][0-9]{0,8}

- **doi**
  - Required: No
  - Type: DOI
  - Pattern: 10.[0-9]{4,9}[A-Za-z0-9:;)(_/.-]+
- **type**
  - Required: No
  - Type: ENUM
  - Allowed values: (see Curators Guide) - `Primary` - `Benchmark` - `Review` - `Other`
- **version**
  - Required: No
  - Type: String
  - Restrictions: Min length: 1, Max length: 100
  - Pattern: [p{Zs}A-Za-z0-9+.,-_:;()]*

Example

```
# XML
<publication>
 <pmcid>21959131</pmcid>
 <pmid>21959131</pmid>
 <doi>10.1038/nmeth.1701</doi>
 <type>Primary</type>
 <version>4.0</version>
</publication>

# JSON
"publication":
[
  {
    "pmcid": "21959131",
    "pmid": "21959131",
    "doi": "doi:10.1038/nmeth.1701",
    "type": "Primary",
    "version": "4.0"
  }
]
```

Example

```
# XML
<publication>
 <isAvailable>Not available</isAvailable>
</publication>

# JSON
"publication":
[
  {
    "isAvailable": "Not available"
  }
]
```

Note:

- if a publication is known to *not* be available, this can be specified using the `isAvailable` attribute (see Example)
- see the curation guidelines.

9.2.23 Credit

Individuals or organisations that should be credited, or may be contacted about the software.

Attribute name credit

Required No

Type List of credit objects

Credit object definition

Content

- **name**
  - Required: Yes
  - Type: String
  - Restrictions: min length: 1, max length: 100
- **elixirPlatform**
  - Required: No
  - Type: ENUM
  - Allowed values: (see Curators Guide)
    - `Data`
    - `Tools`
    - `Compute`
    - `Interoperability`
    - `Training`
- **typeEntity**
  - Required: No
  - Type: ENUM
  - Allowed values: (see Curators Guide)
    - `Belgium`
    - `Czech Republic`
    - `Denmark`
    - `EMBL`
    - `Estonia`
    - `Finland`

* France
* Germany
* Greece
* Hungary
* Ireland
* Israel
* Italy
* Luxembourg
* Netherlands
* Norway
* Portugal
* Slovenia
* Spain
* Sweden
* Switzerland
* UK

- **orcidId**
  - Required: No
  - Type: String
  - Restrictions: pattern: http://orcid.org/{[}0-9]{4}-[0-9]{4}-[0-9]{4}-[0-9]{4}
- **gridId**
  - Required: No
  - Type: String
  - Restrictions: pattern: grid.[0-9]{4,}.[a-f0-9]{1,2}
- **email**
  - Required: No
  - Type: Email
  - Restrictions: pattern: [A-Za-z0-9_]+([-+.'][A-Za-z0-9_]+)*@[A-Za-z0-9_]+([-.][A-Za-z0-9_]+)*.[A-Za-z0-9_]+([-.][A-Za-z0-9_]+)*
- **url**
  - Required: No
  - Type: URL
  - Restrictions: pattern: http(s?)://[^s/$.?#].[^s]*
- **tel**
  - Required: No
  - Type: String

  - Restrictions: min length: 5, max length: 50

- **typeEntity**
  - Required: No
  - Type: ENUM
  - Allowed values: (see Curators Guide)
    * `Person`
    * `Project`
    * `Division`
    * `Institute`
    * `Consortium`
    * `Funding agency`
- **typeRole**
  - Required: No
  - Type: ENUM (list)
  - Allowed values: (see Curators Guide)
    * `Developer`
    * `Maintainer`
    * `Provider`
    * `Documentor`
    * `Contributor`
    * `Support`
    * `Primary contact`
- **comment**
  - Required: No
  - Type: String
  - Restrictions: min length: 10, max length: 1000

Example

```
# XML
<credit>
 <name>TN Petersen</name>
 <orcidId>http://orcid.org/0000-0002-1825-0097</orcidId>
 <gridId>grid.5170.3</gridId>
 <email>test@cbs.dtu.dk</email>
 <url>http://cbs.dtu.dk</url>
 <tel>12345678</tel>
 <typeEntity>Person</typeEntity>
 <typeRole>Developer</typeRole>
 <typeRole>Documentor</typeRole>
 <comment>Lead developer</comment>
</credit>
```

```
# JSON
"credit":
[
  {
    "name": "TN Petersen",
    "orcidId":"http://orcid.org/0000-0002-1825-0097",
    "gridId": "grid.5170.3",
    "url": "http://cbs.dtu.dk",
    "email": "test@cbs.dtu.dk",
    "tel": "12345678"
    "typeEntity": "Person",
    "typeRole":
    [
      "Developer",
      "Documentor"
    ]
    "comment": "Lead developer"
  }
]
```

Example

```
# XML
<credit>
 <elixirPlatform>Tools</elixirPlatform>
</credit>

# JSON
"credit":
[
  {
    "elixirPlatform": "Norway"
  }
]
```

Note:

- a credit consists either simply the name of an ELIXIR Platform or ELIXIR node *or* the name of some other entity that is credited, with associated metadata
- the credit name may only contain space, uppercase and lowercase letters, decimal digits, plus symbol, period, comma, dash, underscore, colon, semicolon and parentheses.
- line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces are not allowed / will be removed.
- see the curation guidelines.

9.3 Entry management attributes

9.3.1 Permissions

Attribute name editPermission

Required No

Type Permission object

Permission object definition

Content

- **type**
  - Required: Yes
  - Type: ENUM
  - Allowed values: - `private` - `public` - `group`
- **authors**
  - Required: No
  - Type: List of usernames

Notes 'authors' only need to be provided when type is set to `group`.

Example

```
# XML

# JSON
"editPermission":
{
  "type": "group",
  "authors":
  [
    "ekry",
    "lukbe"
  ]
}
```

CHAPTER 10

Hangouts

Monthly informal meetings to discuss all matters around bio.tools including ELIXIR EXCELERATE WP1 ("Tools Interoperability and Service Registry") tasks and activities of ELIXIR Denmark technical staff.

You are welcome to attend; please mail Henriette Husum Bak-Jensen (hhu@bio.ku.dk) cc Jon Ison (jison@bioinformatics.dtu.dk), including your gmail and skype addresses. To understand how we organise tasks and projects, read the Contributors Guide.

10.1 2017 Meetings

- No meeting in July, Aug
- POSTPONED (date TBD) Fri Sep 29, 11 AM CEST **WP1-focus**
- Fri Oct 27, 11 AM CET
- Thu Nov 23, 12.30 CET for 2 hours **WP1-focus**
- No meeting in Dec

10.2 Next meeting

10.2.1 2017 Novmber 23, 12.30 - 2.30 PM CET

Call details

Connection details:

1. Plug in your headset **before** joining the call
2. Follow the link: https://c.deic.dk/wp1call/
3. Write your name. Click Participate Now.

Software (Adobe Connect) may need to be installed before entering the meeting room the first time : it's **strongly recommended** you click the link above now, to install the software as necessary.

If you have never attended an Adobe Connect meeting before:

- Test your connection: https://c.deic.dk/common/help/en/support/meeting_test.htm
- Get a quick overview: http://www.adobe.com/products/adobeconnect.html

Attendeees

Representatives of ELIXIR EXCELERATE WP1 partners are expected to attend.

Confirmed

- Jon Ison (DTU, DK)
- Matus Kalas (UiB, NO)
- Dan Søndergaard (AU, DK) + student helpers
- Vassilios Ioannidis (SIB, CH)
- Severine Duvaud (SIB, CH)
- Heinz Stockinger (SIB, CH)
- Veit Schwammle (SDU, DK)
- Henriette Husum Bak-Jensen (KU, DK)
- Ahto Salumets (UTARTU, EE)
- Josep Gelpi (BSC, ES, *provisional*)
- Hedi Peterson (UTARTU, EE)
- Radka Svobodova (MU, CZ)
- Jonathan Hickford (EBI)
- Peter McQuilton (OERC, UK)

Unconfirmed

- Emil Rydza (DTU, DK)
- Hans Ienasescu (KU, DK)
- Lukasz Berger (DTU, DK)
- Piotr Chmura (DTU, DK)

Apologies

- Erik Jaaniso (UTARTU, EE)
- Salvador Capella (BSC, ES)
- Tomas Racek (MU, CZ)
- Salvador Capella (BSC, ES)

- Hervé Menager and/or Kenzo Hillion (IP, FR)

Agenda

1. **Purpose of this call** (Jon Ison)
2. **Deliverables & milestones due in 2018** (Jon Ison)
   2.1. Summary (https://biotools.sifterapp.com/projects/40459/issues?srt=milestone)
   2.2. M1.3 - literature integration (update)
   2.3. M1.6 - metacatalogue + synergy meetings (update)
   2.4. D1.5 - metrics in registry (update)
   2.5. D1.7 - user helpdesk / fora (action needed, see below)
   2.6. D1.8 - matchmaking service : (discussion needed - recommend to drop this)
3. **WP1 partner round-table presentations of what's been done and what's planned for 2018** (5' / partner)
   3.0. Overall priorities and goals for WP1 for 2018 (DTU et al.)
   3.1. ELIXIR-DK (DTU et al.)
   3.2. ELIXIR-EE (UTARTU)
   3.3. ELIXIR-ES (BSC, IRB)
   3.4. ELIXIR-FR (IP, CNRS)
   3.5. ELIXIR-CZ (MU)
   3.6. ELIXIR-NO (UiB)
   3.7. ELIXIR-EMBL (EBI)
   3.8. ELIXIR-CH (SIB)
   3.9. ELIXIR-UK (UOXF)
   3.10. ELIXIR-PT (INESC-ID)
4. **Framing a WP1 f2f meeting** early 2018, round-table contributions
   add your suggested agenda items for the f2f here
5. **Publication opportunities** (Jon Ison)
6. **AOB**

Note: Deliverables & milestones

2017 Q3

- M1.3 - literature integration (POSTPONED from Aug 17)

2018 Q3

- D1.3 - registry release + formats
- D1.5 - metrics in registry (POSTPONED from Aug 17)
- D1.7 - user helpdesk/fora
- D1.8 - matchmaking service (Aug 18)

- M1.1.3 - EDAM release + tooling
- M1.5 - Good Practice Guidelines
- M1.6 - metacatalogue + synergy meetings
- M1.7.2 - novel user interfaces

Note: D1.7 user helpdesk / fora Excerpt from EXCELERATE proposal is below. The idea was indicated as high-priority in the EXCELERATE mid-term review. We need to identify leader(s) for this deliverable.

This task will provide direct and indirect user support to deliver impact for ELIXIR end-users. Direct support will be achieved primarily by leveraging the existing and highly popular user bioinformatics forums (BioStars, BioPlanet etc.). A User-support specialist will patrol such forums and respond to questions in one of four ways:

1. Where resources answering to the Users needs exist in the registry, a link to them in the registry will be provided via our API.
2. Where resources exist in the registry, but the registry API cannot be used to answer the question directly, they will request new features of the API and in so doing drive development of the Query Interface.
3. Where an appropriate resource exists but has not been registered, they will request the appropriate registry curator add it to the registry.
4. Where a registered resource exists that is close, but not quite what is required, they will forward feature requests to the appropriate developers, possibly via the Matchmaking Service (D1.5).

10.3 Archive

10.3.1 2017 June 16, 11 AM CEST

Attendeees

WP-1 partners of which the following were present Anne Wenzel, Emil Rydza, Hans Ienasescu, Jon Ison, Matus Kalas, Piotr Chmura, Severine, Henriette Husum Bak-Jensen.

Apologies

Vivi Raundahl Gregersen, Hedi Peterson, Veit Schwämmle, Vivi Gregersen, Ahto Salumets, Salva , Hervé Menager

Minutes

The goal of todays meeting was to go over the proposed standards for tools entries in bio.tools (see https://github.com/bio-tools/biotoolsSchemaDocs/blob/master/information_requirement.rst). The minutes also offer fundamental concerns – that prompt for consideration before launching the standards Several comments were made at the meeting chat and also issues were brought up. Those can be found here https://github.com/bio-tools/biotoolsSchema/issues/77 and more can be added after the meeting, please. The main points - constructive discussion points and actions points – at the meeting, were the following:

The idea of 'revising the standards on an annual basis' is challenging

Four standard tiers/labels are contemplated (OKAY, GOOD, VERY GOOD, EXCELLENT) that are all of 'acceptable' quality. A fifth label (NEEDS TO IMPROVE) is for entries which lack basic information. Each label is associated with

a set of attributes. The set of attributes required to earn a label – or the list of allowed sub-domains to tick a particular attribute, could in principle be changed – if practical experience shows it would be valuable. And so we envision to revisit, with caution, the set of four (five) standards on an annual basis – with input from the community BUT - by all means, any future change in the standards must not bereave a tool of an 'earned' label, or lead to a 'greying' of an annotation void. Rather such changes should apply to future earning of labels, and be presented in the 'background guide info for curators' for verified-label tools, that now needs more annotation work.

Annotation of Not applicable, None exists, Unknown, and Need Updating

These terms are all valuable information, and should be carefully and individually assigned as annotation options, for all attributes. MK made the point that distinct tool types warrant a distinct set of attributes – in order to avoid numerous 'not applicable' annotation results. It was agreed that MK will draft a matrix (tool types vs attributes) that will help decide if some tool types should indeed be assigned a distinct set of attributes, and if not, at least will help capture the adequacy of annotating 'not applicable' for a given tool attribute.

Annotation metrics – assessing quantitative measures on the quality input

This point was made by MK and wants to assess the registry's total number of annotated information on a given attribute. Other obvious quantitative measures include amount of information (most simply number of JSON/XML nodes); last modified (time since); last new version (time since); last scientific publication (time since). This will help us monitor the overall progress on quality of the registry as a supplement to tracking number of users and number of entries (quantity).

Date-stamps

The annotation 'None exists' should be time stamped, because it may be relevant to update the information. The annotation 'Not applicable' should not be Date-stamped, because it will never be relevant information.

Verification of labels

Several arguments were made for and against a Date-stamped verified label of a given tool. In particular if we're dealing with manual verification of earning a given label: 1) this could be seen as censure, by the developers, which would counteract his/her willingness to simply supply the best possible annotation/information on a given tool, 2) it is labour-intensive and possibly old-fashioned (not Wiki-like), 3) there is a danger of the verification process is of lower quality than the annotation process itself. On the other hand, the end user may better trust a manually verified date-stamped label. We need to consider the need of developers (the best provider of info) and of the end-user (trust issue). Including the possibility for developing a machine-learning-driven autocurator (Action Piotr Chmura). It is possible that the ressources spent of verification were better spent to improve the annotation.

10.3.2 2017, Apr 28 11 AM CEST

Attendees

Vivi Raundahl Gregersen, Anders Halager, Hans Ienasescu, Veit Schwämmle, Søren Brunak, Jon Ison, Frode Pedersen, Mathias Haudgaard, Arne Kratz, Anne Wenzel, Henriette Husum Bak-Jensen

Agenda and Minutes

1) Workplan for importing public domain information on 11.152 tools from MyBioSoftware to bio.tools (HH, 5')https://biotools.sifterapp.com/issues/356

HansI and HH will produce a work plan to ensure a staged import of public domain info on the 11.152 tools from MyBioSoftware, so that the entries will appear progressively month by month, to be completed by end of 2017. They will also plan the curation effort, which must take place in parallel. HH will ask for renewal of volume and quality target for bio.tools entries at the next ELIXIR steering meeting (June XX) as current volume target (10.000 by end of 2017) will be reached ahead of time.

2) Agenda outline for Bio.tools pre-meeting in Odense, August 23 2017 (JI, 10')

This is an open-day meeting for the bio.tools community and all are welcome.

Action: JI will propose a draft agenda by May 5.

3) Bio.tools presentation at Odense Danish Bioinformatics Conference (JI, 5')

Yes, there should be a bio.tools presentation at the conference. The presentation could start by a general update by SB/JI on bio.tools achievements and ambitions, leading on to a talk on bio.tools' scientific purpose: for example invite Magnus Palmblad to present case of using EDAM as basis for guided workflow composition.

Action: JI & Veit to ensure time-slot on conference program (Rikke Stefansen) and once speaker list/titles are confirmed, to mature content of presentation in dialogue with SB.

4) Update on bio.tools content #307: Bioinformatics Links Directory, 621 databases (Ahto Salumets, 5')

This was not covered, but Ahto reports from behind the scenes, that task is nearly completed.

5) 'Regate' as means to harvest tools from local Galaxy servers – an option? Probable number of tools found? Timeline? (Hervé Menager, 5')

This was not covered.

6) CONDA task proposal https://biotools.sifterapp.com/issues/100 , next steps (Dan S, 5')

Three student programmers have started. First task is to create map of existing bio.tools ID's to CONDA ID's and identify un-matched entries in CONDA. The manual work associated with establishing links between CONDA ID's and stable bio.tools ID's must however await the nearly completed cleaning of the bio.tools ID list. The CONDA task fits nicely with the biocontainers project (see **'sifterapp 100 <https://biotools.sifterapp.com/issues/100 >'_**) a container package registry integration effort for container-ised tools found in e.g. dockr and CONDA. A studentship proposal describes in detail, what the CONDA task aims to achieve **'here <https://docs.google.com/document/d/1w31T6w3j0JP7h2Ujp737RhiBcnywiBJ4VNGygdwAdY/edit#heading=h.ok40z7l1xy2h >'_**

7) WP-1 studentsships: new proposals (JI) and status of ongoing ones Proteomics tools annotation (Veit) and Utility to convert open-API configuration files to importable files (Herve) 15')

The work on the proteomics tools annotation is progressing well since it started 3 weeks ago. Hervé could not attend this meeting due to a conflicting ELIXIR meeting. HansI and HH are recruiting on 3 studentships to assign publications on entries without a tool-specific publication or citation or proxy paper. Entries without any of the former curation will be subject to decision if to keep or delete from registry. SB made the point that publications, alt-metrics, number of citations, de-duplications and consistent EDAM assignments, are key curation targets. In parallel, interface functionalities and search functions should be enabled on the development side, to make the most of this entry-information.

Proposed action: JI to please consider if ROADMAP reflects SB's point above, and with what timeline, and share the plan on next SG meeting in June.

8) AOB

None.

10.3.3 2017, Mar 31 11 AM CEST

Attendees

Anne Wenzel, Emil Rydza, Hans Ienasescu, Jon Ison, Veit Schwämmle, Vivi Raundahl Gregersen, Salvador Capella-Gutierrez, Henriette Husum Bak-Jensen, Anders Halager, Dan Søndergård,Jaroslaw Kalinowski, Matus Kalas, Mikkel Schierup

Apologies

Hervé Ménager, Vassilios Ioannidis

Agenda and Minutes

Ad 1) EXCELERATE WP 1 mid-term report (JI, 5 min). The 1st EXCELERATE WP1 periodic report was submitted on 31 march. It will be subject to scrutiny at the April mid-term ELIXIR review. The report is a reference document that compiles the work done so far on WP1. It is recommended reading for everyone involved on WP1, to get up to speed.

Ad 2) Urgency of bug fixes in preparation for a) EXCELERATE mid-term review, b) indexing of Tool Cards, c) in 2017 Q3 the "pivot to end users" (JI, 10 min).

The DTU/KU team of Jon, Emil, Lukasz, and Piotr can handle the urgent tasks that needs doing before the mid-term review. We're all encouraged to take a critical look at bio.tools and give feed-back via github on what we think is the most broken. Salva (ES) mentioned they will contribute a developer to this effort. On this note, please observe that github is the tracker for raising fine-grained issues/critique, while Sifter is used for high-level project management, while the Roadmap addresses the question of 'when' planned bio.tools technical software development will happen. **Action for JI**: to priority-label comments made in github in accordance priority-labelling used in sifter app (i.e. critical, high, normal, low, trivial) to acknowledge the community effort of raising issues in github. Toolcards are about to be indexed in preparation for the coming 'pivot to end-users' task.

Ad 3) Introducing WP1 team from Aarhus Univ + options for WP1-EXCELERATE Milestone assignments (Mikkel Schierup, 10 min). A warm welcome to the WP1 team from AU, presented by Mikkel Schierup. The team is constituted by Anders Halager, Jaroslaw Kalinowski and Dan Søndergaard + three student programmers (10 hrs per week from April).

CONDA task proposal (Dan Søndergaard and Anders Dannesboe) CONDA is 'the standard' open source software package manager. Bioconda is a 'channel' that already contains >3600 bioinformatics-related packages, that is maintained and expanded by a 'serious' open-source community (ContinuumIO). The AU-team proposes a task with the goal of making the maximum number of packages from bio.tools available as Conda packages, and distribute these via Bioconda. Furthermore, they propose to make Conda the official bio.tools approach for installing bio.tools curated software (i.e. bio.tools to inform/educate the end-user on how to install and update packages on different platforms via Conda/Bioconda). Several benefits could arise from such a collaboration including an improved search mechanism on bio.tools and improved understanding of end-users needs. Also, it would give bio.tools a competitive edge. **Conclusion:** The idea is great, and should be written up as one or more studentship-like proposals (see next point) that also addresses the aspect of whether to include packages of single tools and workflows and the boundaries we then would share with parallel ELIXIR activities in the Biotools roadmap. **Action for JI and Dan** to shape project(s) via dialogue in sifter task #100: Support pull of data from content providers.

Sifter tasks proposals The AU-WP1 team also proposed to contribute to sifter apps 240 (Expose bio.tools for indexing by Google), 106 (Enable sorting by citation rate matrics combined with recent citations somehow) and 239 (field for content reviewed), which is warmly welcomed and much appreciated.

Ad 4) WP1-Studentships. Frame and how to apply for these + studentship proposals already made (HH+JI, 10 min).

The Danish ELIXIR node has allocated funds for WP1-studentships. Only curation-focused mini-projects with a clear and quantifiable impact on bio.tools content will be considered for funding. In order to apply for a studentship, a one-page proposal must be written and submitted in accordance with the guidance found here. Generally, a studentship is equivalent to maximum one month of full-time employment. Each project should target producing a mini publication and the project progress towards goals must be tracked in sifter. until now, two studentships have been granted with supervisors Veit Schwämmle (Proteomics tools annotation) and Hervé Menager (Utility to convert open-API configuration files to importable files), respectively. **Action point for Veit and Hervé:** please create sifter tracking for your studentships progress prior to next hangout.

Ad 5) Recent discovery by Hans of 'MyBioSoftware portal' of 11.152 tools timeline for import to bio.tools (Tomas Racek/Jon Ison 5 min).”” Tomas Racek was invited with short notice, and could not join this call. The discovery and work this far is described here sifter task 356.

Action for Jon and Tomas: A timeline and work plan for importing the tools found in MyBioSoftware into bio.tools at standard annotation quality, is needed for the next hangout + the discovery of MyBioSoftware should be added to the monster list. **Action for HH:** The discovery calls for a revision of KPI targets.

The remaining points could not be covered in time, and were postponed for the next hangout on April 28

10.3.4 2017, Jan 27 11 AM CET

Attendees

Anne Wenzel, Emil Rydza, Hans Ienasescu, Jon Ison, Veit Schwämmle, Vivi Raundahl Gregersen, Hervé Ménager, Kenzo Hugo, Anders Halager, Salvador Capella-Gutierrez, Henriette Husum Bak-Jensen,

Thanks to everyone who managed to join this technically challenged meeting ! It seems that hangouts aren't suitable for meetings of 10 participants or more, and so the next TC (Feb 24, 11:00 CET) will take place in another way (Action Henriette),

Please have a look at the revised (27/1 p.m.!) status report here <http://biotools.readthedocs.io/en/latest/status_reports.html

Agenda and Minutes

Ad 1) Hackathon at Aarhus University Feb 2-3 2017: Outstanding issues (Vivi Gregersen) 10 min

Currently 15 people have signed up to this hackathon, everyone is welcome to attend and can study the program AND register here https://docs.google.com/document/d/1tVemqzmus8BpQxfPZRmh5PGmIe64F9a72OKmPhfz1sk/edit#heading=h.p1b4r4t4pje3 Jon will share a spreadsheet template with Vivi, to help define conceptual workflows, relevant tools and annotation (Action Jon) Hans will demonstrate the Tool Annotator as requested – Jon should give directions to Hans as to timing and duration of this (Action Jon).

Ad 2) Status on RTH - RNA tools (Anne Wenzel) 5 min

The upload of ~400 tools that were scheduled for end 2016 has been paused by RTH. This is due to concerns from RTH, as to how the ontology helps in finding the right tools, caused both by limitations in search function support and a non-implemented EDAM ontology extension that RTH plan to do. Anne, Emil and Jon will address these concerns off-line, update the list of critique points to address re: registry developments here https://biotools.sifterapp.com/issues/317 and identify a new plan for uploading the tools, involving Jan Gorodkin (Action Anne).

Ad 3) Tool Annotator – status (Hans Ienasescu) 10 min

The Tool Annotator is currently not integrated with bio.tools but it will be after user feed-back on the current version, at the hackathon in Aarhus Feb 2-3. Here the participants will compare and critique the difference in annotating using the Tool annotator, the bioportal and the current function in bio.tools and Hans will harvest the best modus and upgrade the Tool Annotator accordingly – and then settle on a plan, with Emil, Jon, to integrate it with bio.tools (Action Hans)

Ad 4) Experience from Proteomics workshop Bio.tools outreach (Veit Schwämmle) 10 min

Approximately 30 people attended the workshop. These were both Ph.D. students, postdocs and senior researchers. The main outcome was outreach i.e. to introduce ELIXIR and the bio.tools registry to the proteomics community. Another outcome was to define workflows in proteomics analysis, which is useful not only to the registry but also to the ELIXIR training platform, who attended as well (Niall Beard). The event could not have taken place without the ELIXIR-DK financial support, which was a little hard to come by. ELIXIR DK would benefit from an operational strategy that lowers the bar on resource decisions and executing these (Action Henriette).

Ad 5) Highlights from 'User feedback from the UI tests' see here (Kenzo Hugo Hillion) 10 min

Several constructive points of critique were raised by the report. Salva also raised important points at this meeting. Jon and Emil are grateful for this helpful critique and kindly request these be noted in the sifter task here https://biotools.sifterapp.com/issues/317 where they will action them (i.e. link them with the roadmap) and solve them as soon as possible/feasible. Again – everyone is welcome (and needed) to help solve these issues – please coordinate with Jon, Emil.

Ad 6) Access to the code repository (Hervé Ménager) 10 min

As a solution to some of the remaining software-level issues of bio.tools, HM and KHH have requested an access to the code repository for bio.tools. That would potentially enable to provide quickly corrections to some of the interface bugs for instance. JI would also like to get this access, in order to contribute to tasks such as QC. ER will provide this ASAP (week of jan. 30th).

Ad 7) New curator in DK (yea!) – roles and tasks, inspirational 5 min

Hans Ienasescu has been hired at UCPH, Bioinformatics Centre, for 1 year as of Feb 15, 2017 as a full-time registry curator. Due to time constraints, this point has been postponed for the next meeting.

Ad 8) AOB None

10.3.5 2016, Nov 25 11 AM CET

Attendees

Anne Wenzel, Emil Rydza, Vivi Gregersen, Henriette Husum, Josep, Emil Rydza, Hervé Manager, Hans Ienasescu, Kenzo Hillion, Josep Gelpi, Vivi Gregersen, Henriette Husum

Apologies

Anders Dannesboe, Lukasz Berger, Jon Ison, Veit Schwämmle, Piotr Chmura, Christian Anthon

Our current primary focus is content, the secondary focus being quality of the content in bio.tools Current #entries 2664 # affiliations 145. 2016-Q4 target is 5000 entries.

Agenda / Minutes:

Ad 1) Welcome everyone - especially to Hervé, Kenzo and Josep - brief sharing of plans regarding content expansion and more Kenzo joined Hervé's team recently and will be focusing on the workbench integration enabler component for e.g. galaxy. Content-wise, Kenzo will be loading ~30 highly curated entries authored by Institute Pasteur on to Bio.tools and sponsor community engagement. Kenzo wishes to contribute to software development and is invited to do so by e-mail to registry-support@elixir-dk.org (John Ison, Emil Rydza, Lukasz Berger, Peter Løngren) in the first instance, with an option to set up a more formal structure if necessary.

Ad 2) KPI monitoring: entry growth curve and contributors growth curve #72 (Emil Rydza, 2016-Q4)

Good progress: The two curves have been constructed and will be made visible in November, here https://bio.tools/stats

We will consider posting other statistics e.g. growth in number of users and number of views, when we launch the registry to enable community engagement.

Ad 3) Settle on 'minimum information for content import to staging area #293' - any further input? (Henriette)

We confirmed the following as the minimum information:

- Name
- Homepage

- Description
- EDAM Topic/descriptors

Additional information will be welcome but given default values i.e. not necessary/possible to fill in:

- Publications
- Type of service

ADDENDUM Jon Ison 28/11/16

Concerning the minimum information requirement for "beta" entries, see https://github.com/bio-tools/biotoolsSchema#information-requirements:

- name
- toolID
- homepage
- description
- tool type
- topic
- function

topic and function can be assigned semi-automatically using edamMap and could default to "Topic" and "Operation" if necessary (undesirable).

All entries labelled as "beta" initially until manually inspected.

ACTION: Jon & Emil to firm up validation / inforrmation requirement for labelling ("beta", "standard", "validated" etc.)

end of addendnum

Anders Dannesboe is assuming a new position on Dec 1 and is nearly done with a script to transfer spreadsheets including tools for mass-import to XML - Anders will handover this task to be finalised/implemented by Jon and Hans for task #107.

Jon should please close task 293 and release full steam on task #107

Ad 4) Status and plans concerning implementation of the staging area for mass-import and 'easy' community-driven content expansion #107 (Emil Rydza, 2017-Q1)

Not discussed in absence of John. It's not clear if John or Emil is leading this critical task – please clarify between you.

Ad 5) RNA tools upload progress #62 and (Anne, Q4-2016)

On track. 380 tools expected to be loaded onto bio.tools. Anne will discuss the RNA ontology list with Josep.

Ad 6) MBG proposal for Bio.tools hackathon on crop and wild-stock tools and databases #178 (Vivi, milestone not assigned)

The date for this hackathon has been settled for 2.-3. February 2017 and will take place in Aarhus, Denmark. Henriette will look for budget coverage. Vivi and colleagues will continue to work to specify the conceptual workflows involved.

Ad 7) Issues on settled milestones - needs for revision ? (all)

None

Ad 8). Carry forward input concerning upcoming WP1/ELIXIR-DK partners TC on Dec 2nd at 10 a.m. UK / 11 a.m. DK

None

Ad 9) AOB

None

Next meeting will take place on January 27, 2017 (as December 30 is cancelled)

10.3.6 2016, Oct 26 11 AM CET

Attendees

Anne Wenzel, Emil Rydza, Hans Ienasescu, Jon Ison, Veit S,Vivi Gregersen, Henriette Husum

Apologies

Anders Dannesboe, Christian Anthon, Lukasz Berger, Piotr Chmura

Agenda / Minutes:

Ad 1) Plan for bio.tools content expansion (Jon Ison)

We currently have ~2700 entries in bio.tools and - assuming additions in 2016 Q4 occur as scheduled - are about on track with the registry growth targets in the top down plan which are:

- 2016 Q4 5000 entries
- 2017 Q1 6250 entries
- 2017 Q2 7500 entries
- 2017 Q3 8750 entries
- 2017 Q4 10000 entries

In the current phase, the primary focus is content, the secondary focus being quality of the content. With this in mind, we decided on two tasks:

Task 1: Mass-import - (assigned to Emil & Jon to complete by Q1-2017):

1. to define the minimum information required for a bio.tools mass-import that would result in a 'beta-version' entry in bio.tools.
2. to device a technical solution to implement this task.
3. to identify candidate collections suitable for import en masse
4. Immediate action: Emil and Jon to track this task in sifter.

Jon Ison note (1/11/2016)

- https://biotools.sifterapp.com/issues/107
- https://biotools.sifterapp.com/issues/107
- https://biotools.sifterapp.com/issues/295

Criteria for mass-import task solution:

- Minimum information includes at least Name; website; short description; EDAM descriptors
- The author/owner of the mass-imported tool must be notified by e-mail upon mass-import with guidance to qualify the content to production version.

Task 2: Student helper – minimal annotation (assigned to Veit to complete with Jon by Q4-2016):

1. to revisit the idea of minimal annotation of bio.tools content and define the minimum information required for a beta-version entry to upgrade to production version.
2. to write an instruction for student helpers (and for authors/owners see mass-import task) to perform the required annotation.
3. to present a plan for distributing the annotation task by student helpers across the Danish partners.
4. immediate action: Veit and Jon to track this task in sifter

Jon Ison note (1/11/2016)

- https://biotools.sifterapp.com/issues/294

Ad 2) Sifter app tasks: Are milestones set - questions in this regard (All)

Milestones for all sifter app tasks (except IDEAS) should be assigned and agreed on Jon Ison. Please keep an eye on your milestones and report at hangout meetings, if you want to change the assigned milestone.

Ad 3) MBG proposal for bio.tools hackathon on crop and wild-stock tools and databases (Vivi)

MBG wishes to host an international hackathon in w5 or w 11, 2017, which is great. We will discuss the concrete plans at the next hangout meeting on Nov 25. For that, Vivi will reach out to relevant others and

- define the conceptual workflows for research in the field, which will help to form work-groups at the hackathon, to develop EDAM ontology, as well as expand the list of tools/databases for import, which currently counts ~250 entries. Practically, up to 50 people can attend the event. -
- settle the date for the event by doodle to the registry core list, EDAM core list and this forum.
- settle the location for the event (which could be co-located to other relevant scientific event)
- draft a budget outline for the event

Ad 4) RNA tools upload progress and emerged EDAM ontology issues (Anne)

The plan to upload ~400 RNA tools in 2016 is on track. EDAM ontology challenges have emerged, as pointed out by Jan and Anne by email/progress report. Jon mentioned the opportunity to use synonyms for semantic enrichment of the EDAM ontology, and that some keywords can go to 'operations'. Anne should send the ontology suggestions to Jon I, who will help making the EDAM vocabulary match the need from RNA tools field.

Ad 5) AOB no issues were discussed.

10.3.7 2016 Sep 30 11 AM CET

Attendeees

Anders Dannesboe; Christian Anthon; Lukasz Berger; Emil Rydza; Jon Ison, Henriette Husum

Agenda / Minutes

We deviated from the agenda and focused on the main issue raised by Jon : bio.tools content growth must happen faster. More tools and databases need to be loaded to bio.tools and this must be a critical focus until 1) we are on track with it and 2) practical content growth plan that has been endorsed by the Steering Group. To this end - we will consider the following actions to gear sifterapp:

- complete "top down" anaylsis of curation requirements + ELIXIR EXCELERATE WP1 deliverables and milestones due in 2017 (Jon)

- firm-up practical KPIs, metrics for assesment and propose sensible targets. Map upload targets for WP1 partners & Danish Elixir DK satellite partners (Jon & Henriette)
- map requirements (curation and for milestone & deliverables) to available resources in DK + WP1 partners (Jon in 1st instance)
- assign milestones (i.e. month-year completion needs) to all sifter tasks in "bio.tools content" tracker, this should reflect upload targets for WP1 partners & Danish Elixir DK satellite partners (Jon in 1st instance)
- clarify purpose of planned 'events' and how these each relate to KPI growth (Jon & Henriette)
- prioritise tooling that is essential for content growth, notably the 'moderation interface' (for mass content imports), 'sandbox' functionality (for intermediate registrations) and tool annotator
- organise a f2f meeting for the DK technical group and WP1 partners : 'content growth tactics' sign-off meeting early December 2016, coinciding with the big release (Jon & Henriette)

Henriette and Jon will continue the discussion off-line and come back by email.

Our next meeting is 28 October 2016 from 11:00 DK-time.

10.3.8 2016 July 1 11 AM CET

Call details

Hangouts - Jon initiates

Attendeees

Jon, Henriette, Veit, Anders

Agenda

1. *TASKS* : round-robin catch-up, people say what sifterapp they're working on, asking for help on tasks, reassignment of tasks, etc.
2. *FOCUS* : one person leads a presentation and discussions on a specific point.
3. *STATUS* : people are asked to review the Status Report http://biotools.readthedocs.io/en/latest/status_reports.html before the meeting and bring any points for discussion here, including points from partner institutions.
4. *PRIORITIES* : people are asked to review current priorities on sifterapp, for discussion here.
5. *EVENTS & DEADLINES* : people are asked to bring up items to be actioned in sifter
6. *KPIs* (Emil): Track status of key performance indicators from https://bio.tools/stats. *User accounts* (affiliations); *Recurrent users* (recorded?); *Entries*; *Content changes/edits* (recorded?); *Publications* (bio.tools technical progress - ideas for future publications - what's in progress (sifterapp)
7. *Update on agreed actions* :*Action* Henriette will contact Bernt Guldbrandsen for a representative from AU, QCG for the next meeting (DONE, see Ad 1 below)
8. *What else?* -Program for DKBC pre-meeting/hackathon in Odense (Jon)

Minutes

Ad 1) JI has made posters on ELIXIR, ELIXIR-DK, Computerome, Bio.tools to be presented at ISMB, ECCB, DKBiC and more. Action: JI to please share the posters with the ELIXIR-DK partners and this forum. HH suggests ELIXIR-DK to define national strategy, including sub-strategy for Training and Outreach (Bio.tools-centered strategy for 1) Training Developers, 2) Training strategic segments of end-users in select tools and databases 3) Web-site communication of Danish training events and opportunities. Action: HH to raise issue at next Steering Group meeting (Sept 20th-2016) and to first get input from this forum at the 24 August technical meeting, Odense.

Ad 8) The Elixir Bio.tools OPEN DAY meeting will take place on August 24, the day before the DKBiC meeting. The agenda is found here https://docs.google.com/document/d/1srFDJF43yPGphP8j11DgseiTkaxs7pHeAcj2WyfzH34/edit# and JI will advertise the meeting broadly, with a reminder to register themselves on a doodle. Ad 8) Next two hangouts (end July and August) are cancelled due to holidays and the Open Day meeting, so we will have the next hangout meeting on Friday September 30th.

10.3.9 2016 May 27 11AM CET

Call details

Hangouts - Jon initiates

Attendeees

Veit S, Anne W, José Maria F, Emil R, Maria Maddalena S, Myhanh N, Jon I, Hans I, Henriette H, apologies from Anders Dannesboe

Agenda

1. *TASKS* : round-robin catch-up, people say what sifterapp they're working on, asking for help on tasks, reassignment of tasks, etc.
2. *FOCUS* : one person leads a presentation and discussions on a specific point.
3. *STATUS* : people are asked to review the Status Report http://biotools.readthedocs.io/en/latest/status_reports.html before the meeting and bring any points for discussion here, including points from partner institutions.
4. *PRIORITIES* : people are asked to review current priorities on sifterapp, for discussion here.
5. *EVENTS & DEADLINES* : people are asked to bring up items to be actioned in sifter
6. *KPIs* : Track status of key performance indicators from https://bio.tools/stats *User accounts* (affiliations); *Recurrent users* (recorded?); *Entries*; *Content changes/edits* (recorded?); *Publications* (bio.tools technical progress - ideas for future publications - what's in progress (sifterapp)
7. *Update on agreed actions* : *Action* Henriette will contact Bernt Guldbrandsen for a representative from AU, QCG for the next meeting (DONE, see Ad 1 below) *Action* Maria Maddalena should please send the deadlines + events weekly alert to this quorum from now on. DONE.
8. *What else?*

Minutes

Ad 1) Outreach to TESS (sifter 140, Henriette): Henriette is helping organise a workshop (Fall, 2016) between Bio.tools and TeSS on how to enable cross-links between the two ressources.

MBG partner involvement (sifter 178, Henriette): Bernt Guldbrandsen will shortly assign a technical member to help the bio.tools expansion (wild stock and plant breeding) and to participate in our meetings.

Training platform (sifter 141, Henriette): It will be valuable to understand which E-learning ressources (online files, videos, slide decks etc) are available from the satellites. Henriette will ask this information from everyone. Hans I is willing to help make a video tutorial on 'how to load tools into Bio.tools' or 'how to get started, using COMPUTEROME'.

Anne Wenzel is in the process of loading 400 RNA-bioinformatics tools onto Bio.tools, and to adjust EDAM ontology accordingly.

Text mining tool (sifter 99, name edamMap, Veit and Jon): This project uses text mining of software descriptions/abstracts/full texts to extract associated EDAM terms. Among other applications, the results can be used for automatic tool annotation.

Workflow generation (sifter 119, Veit and Jon): EDAM provides powerful information to create pipelines for e.g. data analysis involving multiple tools. The study shows how to find applicable pipelines and presents several use cases for the analysis of mass spectrometry data. The work will be presented at ASMS 2016 (mass spectrometry conference) and a paper draft is being prepared.

EDAM Tool Annotator (sifter 46): Improved annotation of tools using EDAM terms. The tool aims to peform a "smart" term search and picking on EDAM in the effort to provide the best exisiting tool annotations; alternatively term suggestions will also be available

Tools used by ELIXIR trainers (sifter 60): finish curration for high-value tools to trainers.

Ad 2) No volunteer today. But great opportunity if needing input/bounce off idea Ad 3) Credits to Emil for expanding the bio.tools statistics to comprise more parameters. The report could perhaps be made to contain the 'priority' dimension (Henriette and Jon to liase before the meeting, about this) ad 4) Not done. We really should. ad 5) Not covered, due to time pressure. ad 6) Henriette will contact Emil about KPIs and tracking these

10.3.10 2016 April 29 11AM CET

Call details

tbd

Attendeees

Agenda

1. Scope & purpose of these hangouts
2. Format
    - *Google hangout ?*
    - *skype ?*
3. Quorum
    - *formal or informal ?*
4. Fixed agenda items
    - discussion of bio.tools status report (Emil and Jon will publish, on the last Thu of each month) including status on key performance indicators:
    - #User accounts

- #Entries
- #Content changes/edits
- #Publications on technical progress
- forthcoming deadlines
- forthcoming events
  - ECCB2016 3-7 Sept 2016
  - ELIXIR-DK technical get-together and bio.tools workshop in one event 24. August 2016
- *what else ?*

Minutes

Ad 1) These hangouts should have a practical focus (defined by fixed agenda items) but in-depth technical discussions should be taken elsewhere. We agreed on a set of fixed agenda items, see under 4.

Ad 2) Google hangout worked well today, and we will use this going forward.

Ad 3) All DK partners are expected to provide a representative to these meetings. Currently, we don't expect representatives from industry partners.

Ad 4) The fixed agenda items were agreed to be the following: 1) *TASKS* : round-robin catch-up, people say what sifterapp they're working on, asking for help on tasks, reassignment of tasks, etc. 2) *FOCUS* : one person leads a presentation and discussions on a specific point. 3) *STATUS* : people are asked to review the Status Report before the meeting and bring any points for discussion here, including points from partner institutions. 4) *PRIORITIES* : people are asked to review current priorities on sifterapp, for discussion here. 5) *EVENTS & DEADLINES* : people are asked to bring up items to be actioned -> sifter 6) *KPI's* : Track status of key performance indicators from https://bio.tools/stats

The fixed agenda items will enable the hangouts to serve three overall purposes 1) To surface if Elixir-DK activities are progressing as planned, and if not, what changes/resources are needed? 2) To surface information/results (from Elixir-HUB, -events, -meetings) that need to go to the DK-partners or to the HUB. 3) The meetings serve as a feeder for Elixir-DK Steering group meetings, and similarly, activities/decisions from the Elixir-DK Steering group can be channeled to the agenda of the hangout meetings

Today's actions were: *Action* Henriette will contact Bernt Guldbrandsen for a representative from AU, QCG for the next meeting (ad 3) *Action* Maria Maddalena should please send the deadlines + events weekly alert to this quorum from now on (ad 4)

Today's KPI records were: #User accounts (affiliations) = 262 #Recurrent users = not sure (not recorded?) #Entries = 2403 #Content changes/edits = not sure (not recorded?) #Publications : bio.tools technical progress - ideas for future publications - what's in progress (sifterapp)

CHAPTER 11

Roadmap

All developments of bio.tools software and content are informed by:

1. Community requests including partners and end-users as tracked on GitHub.
2. Delivering priorities of the ELIXIR EXCELERATE grant (granted in April 2015) including revisions in light of 2017 midterm review.
3. Priorities of the ELIXIR Danish node.
4. Personal priorities of the bio.tools team, having insight of the core requirements.
5. Events on the ground.

For a summary of planned developments including priorities and milestones see https://biotools.sifterapp.com/.

- bio.tools features roadmap
- bio.tools content roadmap

Please join the discussion in sifter and GitHub. The bio.tools core team is tiny, so bug fixes, new features *etc.* take a while - you're patience is appreciated!

For a sifter account mail Jon Ison.

CHAPTER 12

bio.tools Studentships

ELIXIR Denmark - the coordinating node of the bio.tools project - has earmarked funds to support studentships to work on curation-focussed mini-projects for bio.tools. Projects must have clear and quantifiable impact on bio.tools content, in terms of number of entries and / or content quality. Projects can include developments of some tooling, so long as this contributes directly to the project goals.

If you would like to propose a project, then please discuss your ideas first by mailiing Jon Ison cc Peter Longreen and Henriette Husum Bak-Jensen. If following this discussion, we all agree there is basis for a project, then we'd require a 1-page project proposal, the text of which we can work on together and in collaboration with other memebers of the registry-core group. Funding will be prioritised (by Jon, Peter and Henriette) for proposals having the biggest potential impact on bio.tools content and quality.

We anticipate most projects to be short duration (normally the equivalent of a month full time work) however there is flexibility, especially where we find talented students who can clearly demonstrate that there work has made an impact. In case of project continuation, progress would be reviewed, and funding for projects that did not perform would be terminated.

12.1 Requirements

- each proposal requires (at least) two named mentors:
  - someone to vouch for the student, provide local on-site supervision, and handle payment of the student
  - someone (normally from the the registry-core group) who will assist with supervision and oversee the delivery of the work
- students must be enrolled with an accedited University, or have accepted a place at such
- any tooling developed during the studentship would have to made freely available under open license.

12.2 Answers to FAQ

- you are welcome to apply at any time

- there is no limit to the number of proposals, although a student can only be employed on one project at one time
- you cannot participate both as a mentor and a student
- only an individual may work on a project; groups cannot submit proposals
- when writing a proposal, please refer to the existing proposals below and follow the general structure and style
- projects must have clear and quantifiable impact on bio.tools content, but you are free to propose anything to these ends: you will need to inspect https://bio.tools and https://dev.bio.tools (latest dev server) to assess current status
- for futher information, mail Jon Ison cc Peter Longreen.

12.3 Proposals

Finalised proposals are uploaded to https://github.com/bio-tools/Studentships/.

Mining the Scientific Literature for and Annotating Proteomics Software using the EDAM ontology and biotool-sXSD

STATUS: Funded and ongoing. See Proposal. Open for comments.

Harvesting service descriptions for bio.tools using OpenAPI standards

STATUS: Funded and ongoing. See Proposal. Open for comments.

See update on progress

Annotating software tools in a scientific context

STATUS: Funded and ongoing (3 students). See Proposal. Open for comments.

Annotating software tools in domains of the Life Sciences

STATUS: Approved for funding. We are open to proposals from thematic editors. Open for comments.

CHAPTER 13

GitHub projects

Relevant projects hosted on GitHub are shown below.

13.1 biotoolsRegistry

https://github.com/bio-tools/biotoolsregistry

The main bio.tools repo.

13.2 biotoolsDocs

https://github.com/bio-tools/biotoolsDocs

Online docs for bio.tools (these docs!)

13.3 biotoolsSchema

https://github.com/bio-tools/biotoolsschema

Resource description model used by bio.tools.

13.4 biotoolsSchemaDocs

https://github.com/bio-tools/biotoolsSchemaDocs

Online docs for biotoolsSchema.

13.5 biotoolsShim

https://github.com/bio-tools/biotoolsShim

Adapters for converting biotoolsSchema-compatible file formats.

13.6 biotoolsConnect

https://github.com/bio-tools/biotoolsConnect/

Adaptors for content exchange with community projects (SeqWIKI, BioConductor, BioJS, ExPASy, ms-utils.org).

13.7 ReGaTE

https://github.com/C3BI-pasteur-fr/ReGaTE

Content import from Galaxy instances.

13.8 ReMoTE

https://github.com/c3bi-pasteur-fr/ReMoTE

Content import from Mobyle instances.

13.9 OpenAPI-Importer

https://github.com/bio-tools/OpenAPI-Importer

Tool to convert Swagger configuration files to Bio.Tools input XML.

13.10 ToolDog

https://github.com/bio-tools/tooldog

Generate XML template for Galaxy or CWL from the description of tools from the registry.

13.11 biotoolsCompose

https://github.com/bio-tools/biotoolsCompose

Semi-automated workflow design using EDAM-annotated tools.

13.12 EDAM

https://github.com/edamontology/edamontology

Controlled vocabulary used by bio.tools.

13.13 edamMap

https://github.com/edamontology/edammap

Mapping terms and text to EDAM concepts.

13.14 Studentships

https://github.com/bio-tools/Studentships

Proposals for the bio.tools studentship scheme.

CHAPTER 14

Events

We organise many events as well as attend events organised by others. If you want to attend an event or have an idea for an event, please mail registry@elixir-dk.org. As a rule we try to avoid events in July & August. All attendees should please read our code of conduct.

- **Curation Hackathons** ("curatathons") gather providers from across the board to curate their resources, critique the Registry interfaces, and provide a forum for knowledge exchange and collaboration.
- **Thematic Hackathons** engage experts in a specific scientific area to help improve the relevant branches of EDAM, consolidate the existing registry annotations, as well as register new resources within the theme.
- **Resource Hackathons** collaborate with experts from a specific collection of tools and services, typically some other registry, community project or Web portal, to bring the collection up to the ELIXIR annotation standard and expose it in the Registry.
- **Technical Hackathons** focus on ontology, software or other technical developments in support of curation of the Registry, its technical development, applications and integration with other systems.

14.1 Forthcoming events

Technical Hackathon : Towards a comprehensive catalogue of data formats (Autumn 2017 tbd, Amsterdam, NL) *tentative*

A hackathon aimed at providing comprehensive coverage of data formats in EDAM. More details will be added soon.

14.2 Past events

ELIXIR-DK / bio.tools Open Day (Aug 23 2017, Odense, DK)

http://tinyurl.com/registryhackathon14

An informal day of presentations, discussion and hacking around activities of the Danish ELIXIR node, including presentations about the ELIXIR Tools and Data Services Registry (https://bio.tools), bio.tools content and feature development, the EDAM ontology, applications of the registry, future plans and more.

Technical Hackathon : Visual Workflows in bio.tools (Mar 1-3 2017, Tallin, EE)

http://tinyurl.com/registryhackathon13

A three day workshop organised by ELIXIR-EE and partners aiming to implement a proof-of-principle for "visual worfklows" in bio.tools : navigation of bio.tools content with cross-links to TeSS via diagrams for common analytical workflows.

Workshop: The future of proteomics in ELIXIR (Mar 1-2 2017, Tübingen, DE)

https://www.elixir-europe.org/events/strategic-workshop-future-proteomics-elixir

Focussed on creating a white paper to discuss the common infrastructures and services needed by the European proteomics community. bio.tools and EDAM were discussed.

Workshop: ELIXIR discovery portals (ELIXIR Innovation and SME Forum: Genomics and Health - Global resources for local Innovation, Feb 27-28 2017, Helsinki, FI)

The forum was aimed at the companies that use public bioinformatics resources in their business and would like to further streamline this process. The event was jointly organized by ELIXIR Finland, ELIXIR Estonia and the ELIXIR Hub. bio.tools was presented.

https://www.elixir-europe.org/events/elixir-innovation-and-sme-forum%3A-genomics-and-health-global-resources-local-innovation

---

Workshop: bio.tools & EDAM @ 2nd NEUBIAS taggathon (Feb 13-15 2016, Oeiras near Lisbon, PT)

http://eubias.org/NEUBIAS/what-is-taggathon/taggathon-2-gulbenkian-oeiras/

The 2nd NEUBIAS Taggathon hosted and supported by the Gulbenkian Institute of Science, organized by the working group "Webtool" (WG4) of NEUBIAS, and in conjunction with the NEUBIAS training school and the following NEUBIAS conference. We extended the bioimaging sub-domain of EDAM in team work with bioimaging experts, and coordinated the development of biii.info/BISE with bio.tools.

Curatathon : Genomics tools in crop & animal breeding (Feb 2-3 2017, Aarhus, DK)

http://tinyurl.com/registryhackathon12

A curation hackathon aimed at curating software tools used for crop and animal breeding research.

Workshop : bio.tools @ Debian Med Sprint (Jan 12-16 2017, Bucharest, RO)

https://wiki.debian.org/Sprints/2017/DebianMed2017

bio.tools folk join the Debian Med folk for co-hacking and co-learning. We improved EDAM annotations in Debian Med, and progressed towards importing high-quality software information from Debian (Med) to bio.tools.

Thematic Hackathon : Computational Proteomics Resources (Jan 10-13, 2017, Semmering, AT)

http://tinyurl.com/registryhackathon11

A thematic hackathon aimed at curating tools for computational proteomics, co-located with the Computational Proteomics Conference.

Technical Hackathon : bio.tools @ NETTAB : (Oct 24 2016, Rome, IT)

http://www.igst.it/nettab/2016/programme/hackathon/

http://tinyurl.com/registryhackathon10

A one day bioinformatics hackathon organized by ELIXIR held in occasion of the NETTAB 2016 Workshop. The hackathon will include the following two main strands: 1) Biosoftware description using bio.tools and schema.org. 2) Deployment of bioinformatics tools and services through Docker.

Workshop: bio.tools & EDAM @ 1st NEUBIAS taggathon (Sep 14-16 2016, Barcelona, ES)

The 1st NEUBIAS Taggathon hosted and supported by Universitat Pompeu Fabra, organized by the working group "Webtool" (WG4) of NEUBIAS, and in conjunction with the NEUBIAS training school. The aim was to bring-in pre-incubated ideas and elements of the next biii.info/BISE webtool and to progress with its implementation. The presence of bio.tools and EDAM projects ensured coordination of NEUBIAS and EuroBioimaging registry and ontology developments with ELIXIR.

http://eubias.org/NEUBIAS/?page_id=228

Conference: ELIXIR-DK @ ECCB (Sep 3-7 2016, The Hague, NL)

http://www.eccb2016.org/

ELIXIR-DK will have a booth at ECCB and will showcase the work of the Danish ELIXIR node including the ELIXIR Tools & Data Services Registry (dev.bio.tools) and the EDAM ontology.

Conference: ELIXIR-DK @ 2nd Annual Danish Bioinformatics Conference (Aug 25-26 2016, Odense, DK)

http://www.conferencemanager.dk/DKBiC-2016/home.html

ELIXIR-DK will have a booth at DKBC and will showcase the work of the Danish ELIXIR node including the ELIXIR Tools & Data Services Registry (dev.bio.tools) and the EDAM ontology.

Workshop : ELIXIR-DK / bio.tools Open Day (Aug 24 2016, Syddansk Universitet, DK)

http://tinyurl.com/registryhackathon9

An informal day of presentations, discussion and hacking, combining two events in one: 1) ELIXIR-DK staff technical get-together and 2) bio.tools workshop.

Conference: ELIXIR-DK @ IMSB 2016 (Jul 8-12 2016, Orlando, USA)

https://www.iscb.org/ismb2016

ELIXIR-DK will have a booth at IMSB 2016 and will showcase the work of the Danish ELIXIR node including the ELIXIR Tools & Data Services Registry (dev.bio.tools) and the EDAM ontology.

Technical Hackathon : Tools, Workflows and Workbenches (May 18-20, 2016, Institut Pasteur, Paris, FR)

http://tinyurl.com/registryhackathon8

A hackathon bringing together developers from key technical projects from ELIXIR and beyond including: the ELIXIR Tools & Data Services Registry (bio.tools), workbench/workflow projects (CWL, Galaxy, Taverna, Arvados), bioinformatics container solutions and registries, and the EDAM ontology.

Resource Hackathon : ELIXIR-SI Tools & Data Services (Apr 8, 2016, University of Ljubljana, SI)

ELIXIR-SI Registry Hackathon will take place on Apr 8, 2016 12-18h at the Faculty of Computer and Information Science (room PR05). The aim of the hackathon is to register Slovenian Bioinformatics Resources and create a national catalogue of Bioinformatics Tools and Data Services.

Thematic Hackathon : Metagenomics Training Resources (Apr 7-8, 2016, EMBL-EBI, UK)

Organised in collaboration with the GOBLET and the ELIXIR Training Platform.

Resource Hackathon : French Tools & Data Services (Mar 24-25, 2016, Gif-sur-Yvette, FR)

http://tinyurl.com/registryhackathon6

A hackathon bringing together representatives of French bioinformatics communities with the ELIXIR Tools & Data Services Registry, dedicated to the description and cataloguing of French tools and services, to boost their discovery and utility.

Resource Hackathon : Norwegian Tools & Data Services (Mar 16-18, 2016, NTNU Trondheim, NO)

A hackathon bringing together representatives of Norwegian bioinformatics communities with the ELIXIR Tools & Data Services Registry, dedicated to the description and cataloguing of Norway tools and services, to boost their discovery and utility.

Resource Hackathon : bio.tools @ Debian Med Sprint (Feb 4-7 2016, Lyngby, DK)

https://wiki.debian.org/Sprints/2016/DebianMed2016

A resource hackathon focussed on curation and software development towards annotation and registration of tool packages from Debian Med. Annotation of Debian Med packages with EDAM.

Resource Hackathon : EMBL EBI tools (Jan 27-28 2016, EMBL EBI, UK)

A mini-hackathon aimed at curation of EMBL EBI software tools.

Resource Hackathon : de.NBI EDAM Codefest (Jan 19-20 2016, Freiburg Uni., DE)

http://tinyurl.com/registryhackathon7

This hackathon, organised by University of Freiburg, will focus on 1) annotation of de.NBI tools and services, 2) ELIXIR Registry and registration process and 3) Publishing tools in the ELIXIR Registry.

Technical Hackathon : EDAM development heuristics (Dec 1-4 2015, Amsterdam, NL)

http://tinyurl.com/registryhackathon5

This hackathon aimed at preparing EDAM for scaling with registry growth. The focus was to enumerate EDAM development heuristics to ensure usability, identify desirable clean-ups, and to devise quality assurance methods, including usability benchmarking in different scenarios. It also included a thematic session focussing on protein structural biology and the WHAT-IF package.

Curatathon : bio.tools curation (Nov 4-6 2015, Brno, CZ)

http://tinyurl.com/registryhackathon3

The second in the series, will aim for representation in the registry of all ELIXIR nodes, including new partners from Spain, Netherlands, Sweden and Finland, and other key resources beyond ELIXIR.

Thematic Hackathon : RNA analysis (Sep 23-25 2015, Copenhagen, DK).

A thematic hackathon focussed on RNA analysis and seeking to establish an ELIXIR RNA Tools Consortium that the Registry can draw upon in the future.

Thematic Hackathon : defining good practice for resource annotation and registry curation (Aug 23-25 2015, Tallin, EE).

http://tinyurl.com/registryhackathon4

A three day workshop organised and financed by ELIXIR-EE aiming to identify relevant processes and good practice for the annotation and curation of resources for their integration into the emerging ELIXIR infrastructure, focussed on next generation sequencing (NGS) analysis and the SeqWIKI Resource Hub.

Technical Hackathon - EDAM Development & Governance (Mar 11-13 2015, Lyngby, DK)

http://tinyurl.com/registryhackathon2

Focused on EDAM technical maintenance and usability, and produced a mock-up of tooling to assure optimal usage of EDAM for registry curation.

Curatathon - Registration of Tool & Data Services (Nov 19-21 2014, Lyngby, DK)

http://tinyurl.com/registryhackathon

Gathered representatives of institutes and key projects within ELIXIR and beyond. The participants performed a valuable pre-release critique of the Registry mechanism and interfaces, and added more than 300 resources to the content.

Mobyle, EDAM and Service Registry hackathon (Jun 17-18 2014, Paris, FR)

Workshop - ELIXIR, BioMedBridges & RDA Workshop: A common vocabulary to classify resources in the life sciences (Oct 7-8 2014, Brussels, NL)

http://www.biomedbridges.eu/news/workshop-common-vocabulary-classify-resources-life-sciences

---

ALLBIO Workshop - Metagenomics & interoperability (Apr 10-12 2014, Amsterdam, NL)

BioMedBridges AGM Tools Workshop (Mar 9-12 2014, Florence, IT)

---

bio.tools @ Debian Med Sprint (Jan 31-Feb 3 2014, Aberdeen, UK)

ELIXIR/BioMedBridges Workshop on Tool Registries (Oct 16-18 2013, CBS-DTU, DK)

BioMedBridges Registry Workshop (May 8 2013, Imperial College, UK)

AllBio / EMBRACE Continuity Workshop (Mar 18-20 2013, Amsterdam, NL)

BioMedBridges AGM Registry Workshop (Mar 11-12 2013, Dusseldorf, DE)

EDAM hackathon (Oct 9-13 2012, EMBL-EBI, UK)

AllBio workshop - Â¡Â°Web services for improved interoperability in bioinformaticsÂ¡Â± (Oct 2-5 2012, Munich, DE)

14.3 Code of Conduct

We respectfully ask all attendees at meetings to conduct themselves in a way that maintains focus, respect, order - and enjoyment! Suggestions include:

- Bear in mind that you are as responsible for the success of the meeting as anyone else.
- Stick to the meeting agenda if stipulated (most of our meetings do not have rigid agendas).
- Remain focused on the task at hand.
- Come prepared.
- Use an analytic, facts-based approach to problem solving whenever possible.
- Manage meeting time wisely.
- Brainstorm when fresh ideas are in short supply or complex problems present challenges.
- Allow for the expression of every personÂ¡Â¯s ideas, and give all ideas a serious hearing.
- Listen carefully to each other, and be courteous.
- Accommodate disagreements and criticisms without hostility.
- Refrain from all personal attacks.
- Demonstrate flexibility.
- Make meetings enjoyable; employ humour and respect.
- Resolve conflict through compromise and consensus whenever possible.

CHAPTER 15

Governance

bio.tools follows a simple governance model of three tiers under the leadership of the Danish ELIXIR node (Professor Søren Brunak, Head of Node). Development on the ground is led by Jon Ison (technical coordination), Emil Rydza (lead software developer) and Hans Ienasescu (lead curator), in close collaboration with the registry core developers (see below) and EDAM developers.

If you'd like to get involved with the project please mail registry-core@elixir-dk.org.

15.1 Registry Core

Registry Core includes the technical and scientific experts at the heart of the development and curation of bio.tools. Priorities are set in a quasi-democratic way with the **technical coordinator** (currently Jon Ison), **content leader** (currently Hans Ienasescu) or **software leader** (currently Emil Rydza) having the final over these respective areas, where necessary (in so far as this is meaningful). Registry Core members must either be funded, or have the intent and some bandwidth, to support the registry in the long-term. The technical coordinator ensures the Registry Core group and all Contributors are listened to and informed.

Members of Registry Core are responsible for agreeing aims and general good practice. They are expected to advocate bio.tools and (as bandwidth allows) collaborate with one another to help develop the registry software, related technical projects and registry content, *e.g.*:

- add new and improve existing content through collaboration with EDAM Developers
- routine content maintenance including quality control
- work collaboratively within the Curation Task Force (see below) and attend Hackathons
- suggest or implement new features
- develop software for the registry and related technical projects
- evaluate the registry and provide feedback, to ensure the registry software is fit for purpose

The content and software leaders are responsible for reporting software development and curation priorities, and progress, to the ELIXIR-DK Management.

Registry Core will assemble virtually or in person as circumstances dictate, in meetings with open agenda and followed up with actions and notes on key recommendations. All Registry Core members are signed up to the the registry core mailing list.

15.1.1 Registry editors

Named **registry editors** are registry-core members responsible for overseeing coverage and quality in specific thematic areas, e.g.

- evaluating existing coverage (EDAM, tools)
- driving coverage (EDAM, tools)
- liasing with community & leading workshops in their specialist area

See the Editors Guide.

15.2 Registry Contributors

Registry contributors include anyone who makes significant contributions to the registry content or registry-related software, by whatever means, but have none of the responsibilities or expectations of Registry Core.

An important (but voluntary) role of contributors is to function in an **advisory capacity**, *i.e.* review the progress and priorities of Registry Core and advise them on their priorities and how best to achieve the current aims. To these ends, the following actions are welcome: - Read the status reports and changelog and provide feedback on the reported progress and priorities. - Oversee the registry curation and software development and actively offer constructive advice based on their practical experience, requirements and expertise - Advocate the registry to colleagues

The Registry Core will respect this feedback and advice and reflect it in subsequent rounds of registry development and curation. We very much welcome new contributors: for further information please mail registry-core@elixir-dk.org.

15.3 Registry end-users

We particularly welcome input from end-users from the life science community including scientists, technicians and managers from academia and industry: - to test, evaluate and critique the registry software and content - to provide feedback and constructive advice based on their practical experience, requirements and expertise

The Registry Core will respect this feedback and advice and reflect it in subsequent rounds of registry development and curation. Anyone who is considering using the registry - but especially typical scientist / bioinformatician end-users - are welcome to mail registry-core@elixir-dk.org.

CHAPTER 16

Contributors

16.1 Registry Core

- Jon Ison (DTU, DK) - **technical coordination**, lead engineer for biotoolsSchema & EDAM ontology development
- Emil Rydza (KU, DK) - **software development (lead)**, lead developer for bio.tools
- Hans-Ioan Lenasescu (KU, DK) - **curation (lead)**, Web development
- Henriette Husum Bak-Jensen (UCPH - Dept of Biology) - **project management**, studentships
- Piotr Chmura (KU, DK) - **software development**, bio.tools development (back-end)
- Lukasz Berger (DTU, DK) - **software development** bio.tools development (front-end)
- Hervé Ménager (Institut Pasteur, FR) - **workbench integration**, user engagement, ontology & schema development
- Kenzo Hillion (Institut Pasteur, FR) - **workbench integration**,
- Matúš Kalaš (University of Bergen, NO) - **schema & ontology developer**, user engagement, ontology & schema development
- Ahto Salumets (UT, EE) - **curation**
- Tomáš Raček (Masaryk University, CZ) - **curation**
- Alban Gaignard (CNRS, France) **Semantic Web applications**
- Anne Wenzel (RTH, DK) - **curation** (RNA tools)
- Erik Jaaniso (UT, EE) - **software development**, lead engineer for edammap
- Bjoern Gruening (University of Freiburg, DE) - de.NBI & Galaxy integration
- Dmitry Repchevsky (BSC, ES) - Web services & monitoring
- Jacques van Helden (Aix-Marseille University, FR) - advisor
- Dan Bolser (EMBL-EBI, EU) - WIKI integration

- Magnus Palmblad (LUMC, NL) - msutil.org integration
- José María Fernández (CNIO, ES) - benchmarking
- Karel Berka (Palacky University, CZ) - advisor
- Michael Crusoe (Common Workflow Language project) - advisor, CWL integration
- Peter Juvan (University of Ljubljana, SI) - curation
- Rabie SAIDI (EMBL-EBI, EU) - text mining
- Salvador Capella (INB, ES) - benchmarking
- Sebastien Moretti (SIB, CH) - curation
- Severine Duvaud (SIB, CH) - SIB / ExPASy integration
- Tunca Dogan (EMBL-EBI, EU) - text mining
- Wojtek Dabrowski (RKI, DE) - benchmarking

16.2 Registry Core (Registry Editors)

- José Maria Carazo (CNB/CSIC, ES) - **electron microscopy**
- Josep Gelpí (INB / BSC-CSN, ES) - **structural bioinformatics**, benchmarking & tools interoperability
- Juergen Haas (University of Basel, CH) - **protein structural biology**, benchmarking
- Marta Villegas (BSC, ES) - **NLP** and **text mining**
- Veit Schwämmle (SDU-BMB, DK) - **proteomics**, ontology development, bio.tools applications
- Vivi Raundahl Gregersen (Aarhus University, DK) - **agricultural science**
- Carlos Oscar Sorzano (CNB/CSIC, ES) - **electron microscopy**

16.3 Registry Core (tentative)

- Anthony Bretaudeau (INRA - GenOuest/BIPAA)
- Christian Anthon (University of Copenhagen)
- Laura Emery (EMBL-EBI)
- Olivier Collin (CNRS - GenOuest)
- Peter Rice (Imperial College London)
- Priit Adler (University of Tartu)
- Steffen Möller (University of Rostock, DE)

16.4 Registry Contributors

Thanks to the many people who have contributed - if you're not listed below, please let us know!

- Aleksandra Nenadic (University of Manchester)
- Anders Dannesboe (BIRC, DK) - virtualization / container services

- Anthony Bretaudeau (INRA - GenOuest/BIPAA)
- Bjoern Gruening (Uni-Freiburg)
- Bren Vaughan (EMBL-EBI, EU) - EBI integration
- Carole Goble (ELIXIR-UK)
- Chris Morris (STFC)
- Christian Anthon (University of Copenhagen)
- Christophe Blanchet (ELIXIR FR)
- Dan Bolser (EMBL-EBI, UK)
- Daniel Faria (FCG)
- Daniel Kahn (INRA, Lyon 1 University & PRABI)
- Federico Zambelli (CNR-IBBE)
- Frederik Coppens (VIB, BE)
- Gert Vriend (CMBI, NL)
- Gianluca Della Vedova (Univ. Milano-Bicocca, IT)
- Gianni Cesereni (University of Rome "Tor Vergata")
- Giuseppe Profiti (ELIXIR-IT & University of Bologna, IT)
- Gonçalo Antunes (INESC-ID)
- Guy Yachdav (TUM, DE)
- Hedi Peterson (University of Tartu)
- Heinz Stockinger (SIB Swiss Institute of Bioinformatics)
- Helen Parkinson (EMBL-EBI, UK)
- Hervé Ménager (Institut Pasteur)
- Inge Jonassen (ELIXIR NO)
- Ivan Mičetić (University of Padova)
- Jan Brezovsky (International Clinical Research Center and Masaryk university)
- Jiří Vondrášek (ELIXIR-CZ)
- José María Fernández (CNIO)
- Karel Berka (UPOL, CZ)
- Kaur Alasoo (University of Tartu)
- Kristian Davidsen (DTU, DK)
- Kristoffer Rapacki (DTU, DK) - advisor
- Laura Emery (EMBL-EBI)
- Luana Licata (University of Rome "Tor Vergata")
- Ludek Matyska (Masaryk University)
- Manuela Helmer-Citterich (University Tor Vergata, Rome)
- Maria Maddalena Sperotto (DTU, ELIXIR-DK)

- Marie Grosjean (IFB, FR)
- Marie-Paule Lefranc (IMGT, IGH, CNRS, Université de Montpellier)
- Niall Beard (University of Manchester)
- Niclas Jareborg (ELIXIR SE)
- Olivier Collin (CNRS - GenOuest)
- Paola Roncaglia (EMBL-EBI)
- Paolo Romano (IRCCS AOU San Martino IST)
- Peter Juvan (University of Ljubljana)
- Peter Rice (Imperial College London)
- Priit Adler (University of Tartu)
- Rabie Saidi (EMBL-EBI, UK)
- Radka Svobodova (MU, CZ)
- Rafael Jimenez (ELIXIR HUB)
- Rodrigo Lopez (EMBL-EBI)
- Rune Friborg (Birc, au)
- Rune Møllegaard Friborg (BIRC, DK) - virtualization / container services
- Sebastien Moretti (SIB Swiss Institute of Bioinformatics)
- Severine Duvaud (SIB Swiss Institute of Bioinformatics)
- Silvio Tosatto (University of Padua)
- Sofia Kossida (IMGT, IGH CNRS, University of Montpellier)
- Steven Newhouse (ELIXIR EMBL-EBI)
- Tatyana Goldberg (TUM, DE)
- Timothy Karl (TUM, DE) (2remove: another important contact @rostlab)
- Tunca Dogan (EMBL-EBI, UK)
- Vegard Nygaard (ELIXIR NO)
- Victor de la Torre (INB)
- Wiktor Jurkowski (Earlham, UK)

CHAPTER 17

License

The registry content is freely available to all under the Creative Commons Attribution licence (CC BY 4.0).

Access to the registry source code for the front and back-end, can be granted to trusted partners upon request: please email Peter Longreen.

ELIXIR is an open collaboration between equal partners and in this spirit, the bio.tools source code will be opened to a broader public, at the discretion of ELIXIR Denmark Head of Nodes, in due course.

CHAPTER 18

Publications

Hillion K.H., Kuzmin I., Khodak A., Rasche E., Crusoe M., Peterson H., Ison J., Ménager, H. (2017). Using bio.tools to generate and annotate workbench tool descriptions F1000Research 2017 (article). doi:10.12688/f1000research.12974.1

Doppelt-Azeroual, O., Mareuil, F., Deveaud, Kalaš, M., Soranzo, N., van den Beek, M., Grüning, B., Ison, J. and Ménager, H. (2017). ReGaTE: Registration of Galaxy Tools in Elixir *GigaScience*, doi:10.1093/gigascience/gix022

Ménager, H., Kalaš, M., Rapacki, K. and Ison, J. (2016). Using registries to integrate bioinformatics tools and services into workbench environments *Int J Softw Tools Technol Transfer*, doi:10.1007/s10009-015-0392-z

Ison, J. et al. (2015). Tools and data services registry: a community effort to document bioinformatics resources. *Nucleic Acids Research*, doi: 10.1093/nar/gkv1116

Ison, J., Kalaš, M., Jonassen, I., Bolser, D., Uludag, M., McWilliam, H., Malone, J., Lopez, R., Pettifer, S. and Rice, P. (2013). EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats *Bioinformatics*, doi: 10.1093/bioinformatics/btt113

18.1 Citation

If you use bio.tools, please cite:

Ison, J. et al. (2015). Tools and data services registry: a community effort to document bioinformatics resources. *Nucleic Acids Research.* doi: 10.1093/nar/gkv1116

If you use EDAM or its part, please cite:

Ison, J., Kalaš, M., Jonassen, I., Bolser, D., Uludag, M., McWilliam, H., Malone, J., Lopez, R., Pettifer, S. and Rice, P. (2013). EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats *Bioinformatics*, doi: 10.1093/bioinformatics/btt113

CHAPTER 19

Support

For help, support and feedback, please use the following email: - registry-support@elixir-dk.org