Instructions MetaBot demo

Instructions on how to use MetaBot to generating instances automatically using the demonstration file

For general information and instructions on how to install the package, please read the README file.

In the example folder there is a file called "demo_createSpecimen.py". With this script you can create subjects and samples automatically by importing metadata from a macro-enabled template excel file and/or based on subject information stored in a JSON or CSV file.

How to use

To create instances for samples (either tissue sample or tissue sample collections) from subject that are already in the KG editor, you need to following files:

- `specimen_template.xlsm` (macro-enabled template file provided)
- `demo_createSpecimen.py` (python script to create the instances)

Optional:

- `example_subjects.json` (extracted from the query builder, see instructions below)
- `example_subjects.csv` (previously created subjects, stored in a csv output file)

Template .xlsm file:

The template file contains entries to create subjects or subject groups with their accompanying states (i.e. columns A-J) and entries to create tissue samples or tissue sample collections with their accompanying states (i.e. columns K-U).
You do not have to create subjects and samples at the same time. You can only create subjects or only create samples. You could also create samples after you created subjects by either using the output .csv file from the script or by querying the KG (see instructions below).

Note: Required fields are indicated with *.

Put one subject on one row. If more than one sample should be generated for a subject, put each sample on a new row and include the same subject information (this is important to connect the samples to the correct subject).

To create subjects or subject groups (columns A-J):

- `subjectType*:` Choose subject or subject group
- `subjectName*:` Choose the name of the subject or subject group. This will become the lookupLabel in the editor.
- `subjectInternalID:` specify the name, if left empty, no internal identifier will be added.
- `strainName:` Choose a strain that is currently already in the system (these are "semi-controlled terms at the moment"). If you want to add a strain that is not already in the system, leave this blank.
- `strainAtid:` DO NOT fill in this field. This field will be filled in automatically based on the strain name that is chosen.
- `subjectStateNum*:` number of the states for a particular subjects.
- `subjectStateName:` if left empty, states are named as follows: "state-01", "state-02", etc. For any other name, please define the name here and separate multiple names with a comma.
- `biologicalSex:` Enter the biological sex of the subject. If unknown, leave blank.
- `ageCategory*:` Choose the age category from the dropdown list.
- `subjectAttribute:` Choose a subject attribute from the dropdown list. Leave empty if not applicable.

To create tissue samples or tissue sample collections (column B and columns K-U):

- `subjectName*:` Define the name of the subject or subject group that is already in the editor (corresponds to the query of the specimen (see instructions below)) or that is newly created (see instructions above). This is important to link the samples to the correct subject.
- `specimenType*:`
  "tsc" = tissue sample collection,
  "ts" = tissue samples
- `sampleName*:` Choose the name of the subject or subject group. This will become the lookupLabel in the editor. A good naming convention includes the subject ID, e.g. "sub-01" to be able to link the sample to the correct subject; any unique features, e.g. the brain region "layer1"; and the sample type, e.g. "tsc". Following this example, the sampleName would be "sub-01_layer1_tsc".
- `sampleInternalID:` specify the name used for the files related to this sample. If left empty, no internal identifier will be added.
- `sampleType*:` Choose the type of sample from the dropdown list, e.g. tissueSlice or singleCell.
- `region:` Select the brain region the sample is anchored to from a dropdown list. Check SANDS for proper notation. Include everything starting with "AMBA" or "WHS". This will be used to figure out if parcellationEntity or parcellationEntityVersion needs to be used.
- `origin:` Select the origin of the samples. This could be organ (e.g. brain), or cellType (neuron).
- `quantity:` define the number of samples in the collection (only applies to tsc)
- `sampleStateNum*:` define the number of states, value should be at least 1
- `sampleStateNames:` if left empty, states are named as follows: "state-01", "state-02", etc. For any other name, please define the name here and separate multiple names with a comma.
- `sampleAttribute:` define attribute of the sample here: choose between "stained" or "unstained".

For more information about what controlled terms are available, see the wiki.

NOTE: Save the template file as a .xlsx file. This will remove some of the macro-enabled fields and it will notify you of this, but just accept.

Querying subject information for a particular dataset

Metadata that is already stored in the Knowledge Graph can be queried and used to generate tissue samples, so that they can be directly linked to the studied state of the subject.

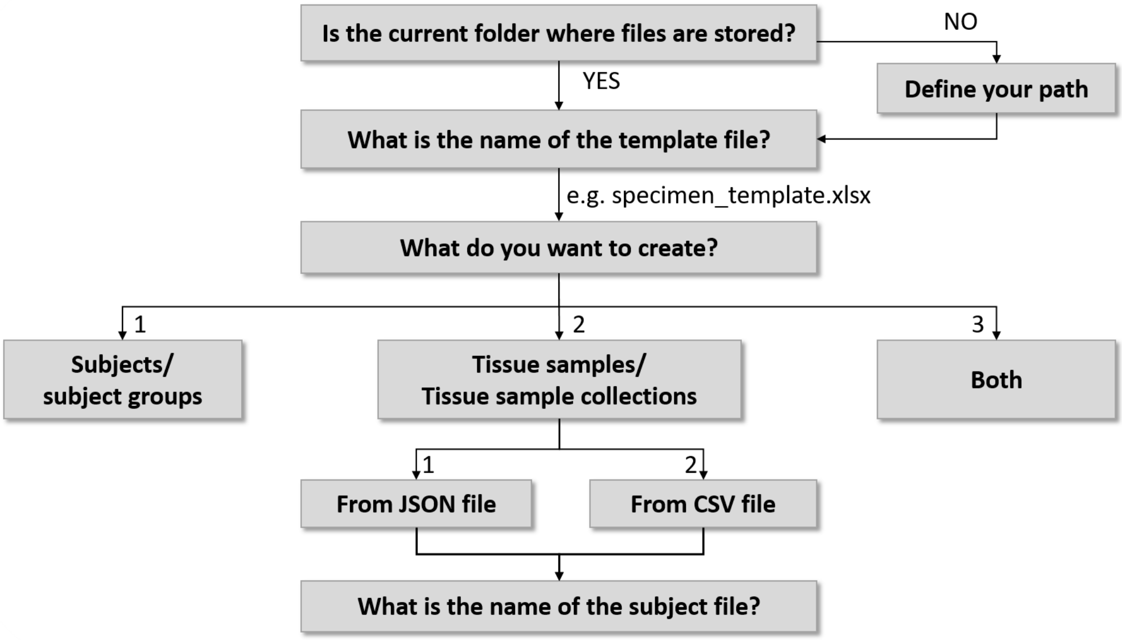
Please note that you will still need to add the tissue samples to the dataset version under studiedSpecimen for them to show up in the KG search UI.

1. Go to the KG Query: https://query.kg.ebrains.eu/
2. Select dataset version
3. Press on the magnifying glass at the top. This will bring you to he saved queries
4. Select the query that is called: `Get-Specimen`
5. Press the button that states "copy as new query"
6. Press on the second line with `* Id > *id*`
7. Copy the UUID of the dataset version that you want to create samples for into the box next to "value" in filter.
8. Go to the play tab.
9. Change "scope" to "in progress", then press run.
10. Once the query has been performed, open the `data`. Only one line should appear with `0:`, because you selected one particular dataset version.
11. Open `studiedSpecimen`. Check if these are indeed the correct subjects.
12. Then hoover at the end of the line with `studiedSpecimen` and an orange "copy to clipboard" icon appears.
13. Copy the information in a text file and save it with the extension ".json", for example: "subjects.json"

Running the python script

You can run the python script from the command line, or from software that allows you to execute python scripts, e.g. spyder (in Anaconda) or Visual studio code.

The easiest is if you put the script in the same folder as the subject and specimen template files, but this is not necessary. The script will ask you a number of questions according to the following decision tree:


Is the current folder where files are stored?
NO
YES
Define your path
What is the name of the template file?
e.g. specimen_template.xlsx
What do you want to create?
1
2
3
Subjects/ subject groups
Tissue samples/ Tissue sample collections
Both
1
2
From JSON file
From CSV file
What is the name of the subject file?

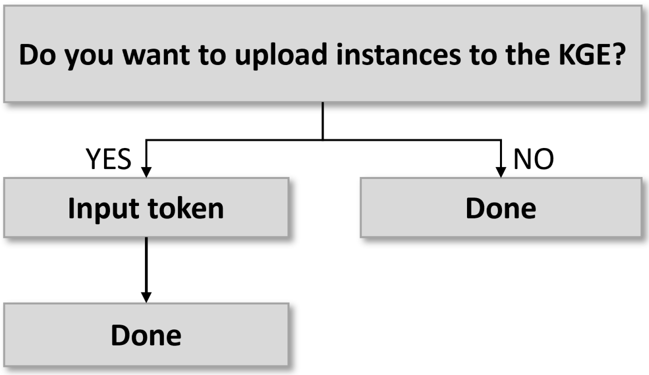
Additional information to answer the questions:

1. "Is this where the files are stored?" < It prints the path to where the python file is located. If all files are in the same folder, press "y". If this is not the case, you can define the path yourself by answering the next question: "Please define you path: ".
2. "What is the name of your specimen info file (e.g. specimen_template.xlsx)? " < Please specify the name of your sample info file. Make sure you saved it as a .xlsx file and do not forget to include the extension.
3. " What do you want to create?" < You can choose to create only subjects/subject groups, only tissue samples or tissue sample collections or you can create both at the same time. Press the following number: subjects/subject groups = 1, tissue sample/tissue sample collections = 2, both = 3.
4. If you choose to only create tissue samples or tissue sample collections, you will be asked whether you want to create it from a JSON file (which is obtained from the KG query for subjects that are already in the system) or to generate it from a CSV file that you previously created by running this script for subject creation.
5. For JSON: "What is the name of your subject file (e.g. subjects.json)? ". Please specify the name of your sample info file. Make sure you saved it as a .json file and do not forget to include the extension.
   For CSV: "What is the name of your subject file (e.g. subjects.csv)? ". Please specify the name of your sample info file. Make sure you saved it as a .csv file and do not forget to include the extension.

The script updates you on which instances are being created and whether information is missing. If information is required and not found in the template file, it tells you to go back to the file and check if the information was entered correctly.

Uploading the instances to the KGE

Once the instances are created, the script asks whether you would like to upload these instances directly to the KGE. To upload the instances to the KGE, you will need to be authorised to do so. If you are, you can use the authentication token generated by the KGE or query builder.



Additional information to answer the questions:

1. "Do you want to upload the instances to the KGE?" < If yes, press "y". You will proceed to the next question. If this is not the case, you can press "n" and your are done. Instances can be uploaded later using the upload function of MetaBot (see ex5.py).
2. "Please copy your input token: " < The authentication token can be found under your account in the KGE or query builder. Pressing on the account symbol in the top right corner of the KGE or query builder, and select "copy token to clipboard". Paste the token as response to this question.
3. When the instances are uploaded to the KGE, the script prints whether it was successful, or whether there are any authetication issues (e.g. error code 401), or whether the instance already exists in the Knowledge Graph (i.e. error code 409).
   If you get a "401" error, please refresh your browser and copy a new token (tokens are only valid for a short amount of time). In this case, do not rerun the demo script, since this will create new instances. Instead, use the example script ex5.py.

Contributors

Maaike M.H. van Swieten (mvanswieten@outlook.com)

License