Hpo Case Annotator Documentation

Release 2.0.0-RC2

Peter N Robinson, Daniel Danis

Feb 15, 2022

Index:

Hpo Case Annotator is an application designed to help biocurate pathogenic variants published in scientific literature. Each case that is curated will contain details of the disease-causing variants, phenotype, disease, and other metadata. Curated data is stored in `JSON` format (one file for each case). The application is also designed to perform a number of Q/C checks to ensure that the data is consistent. The app can export data in Phenopacket format, but it contains a superset of the information required for phenopackets. Future versions of this app will probably converge to the Phenopacket format, and currently the app is still in a preliminary stage of development, although it works as advertized.

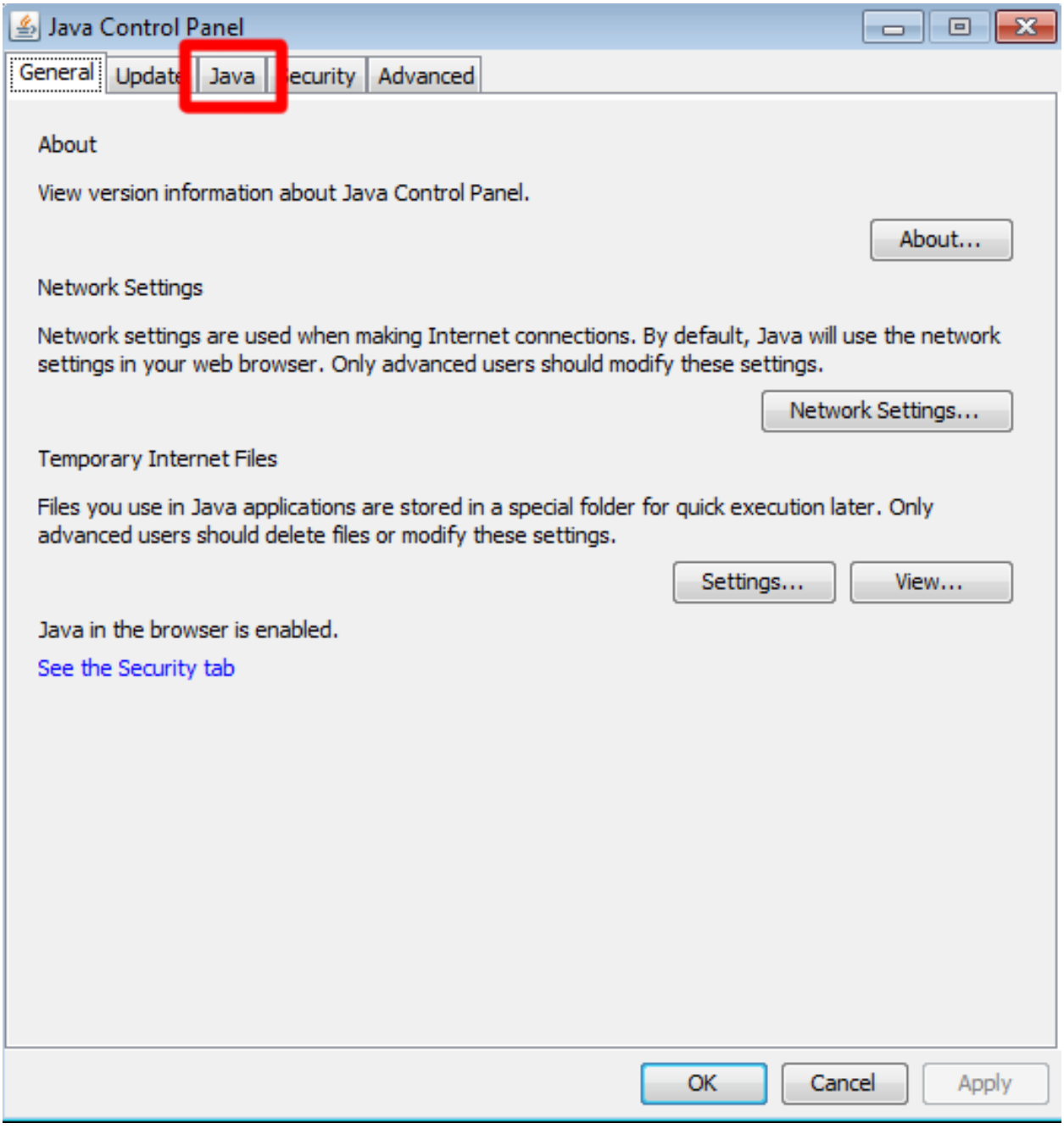
CHAPTER 1

Requirements

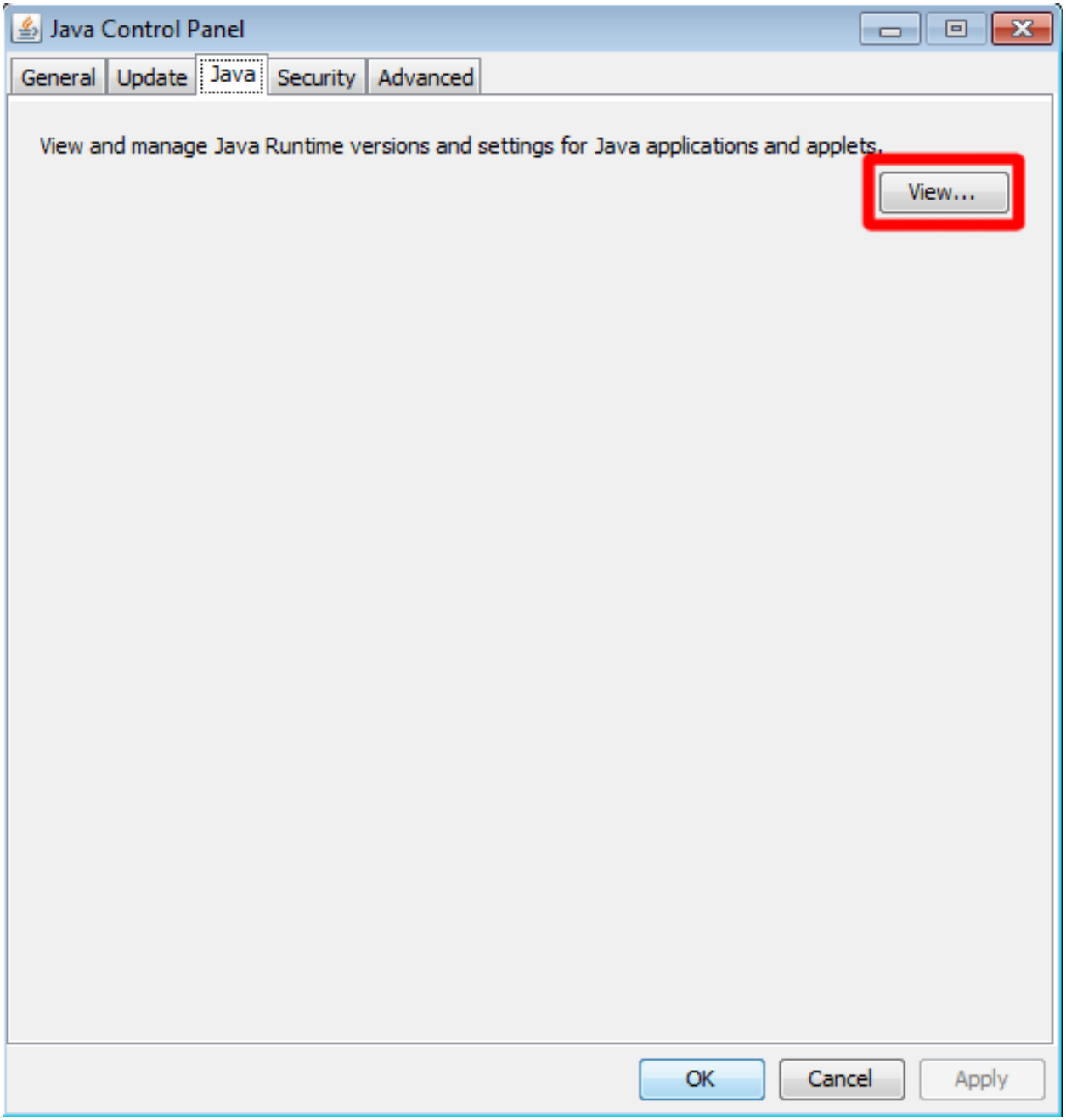
In order to run, *Hpo Case Annotator* needs **Java 8** to be installed on the computer. This page describes steps required to check which version of Java (if any) is installed on your computer.

1.1 Windows

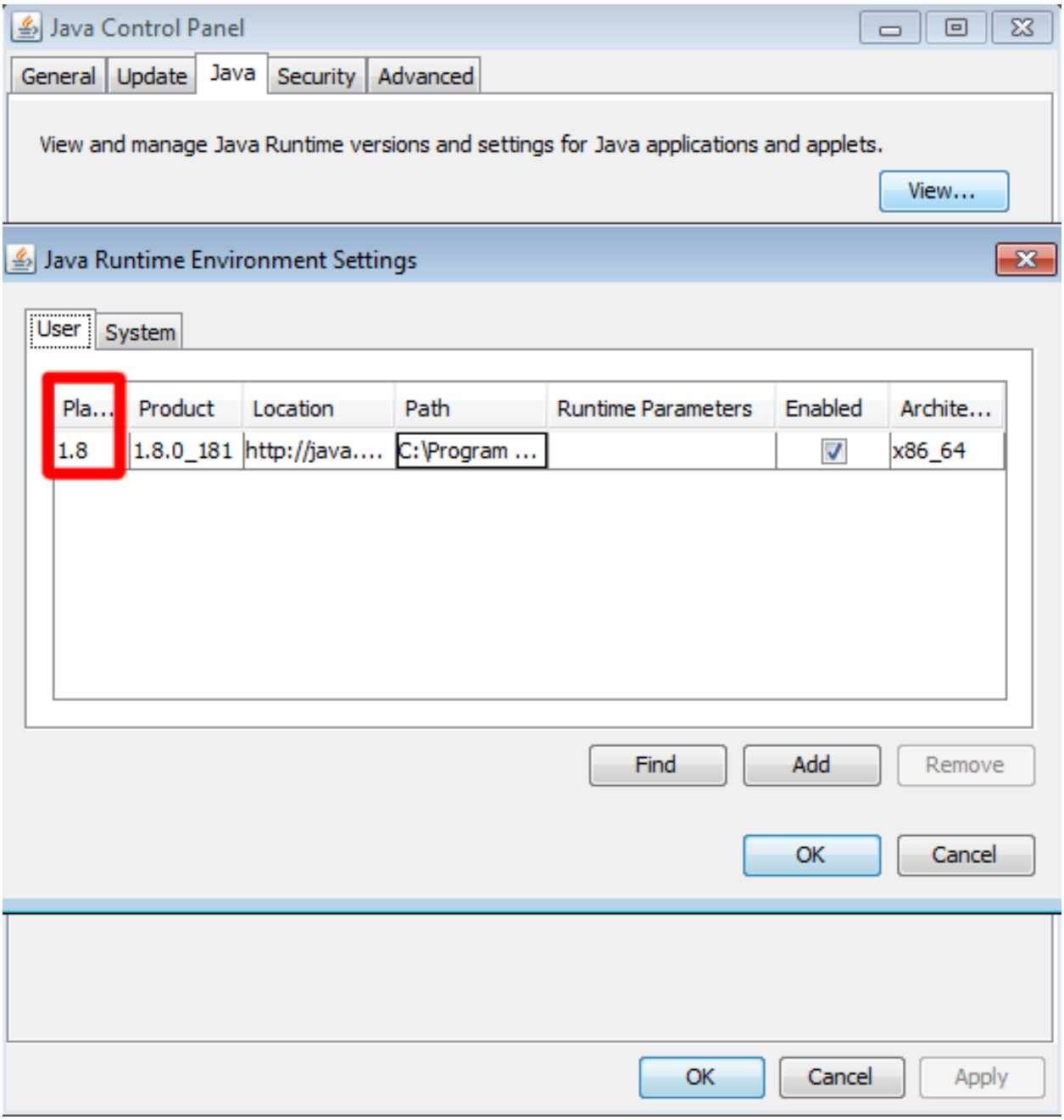
Open **Java Control Panel** and select the **Java** tab at the top of the window.



Click on the **View** button.



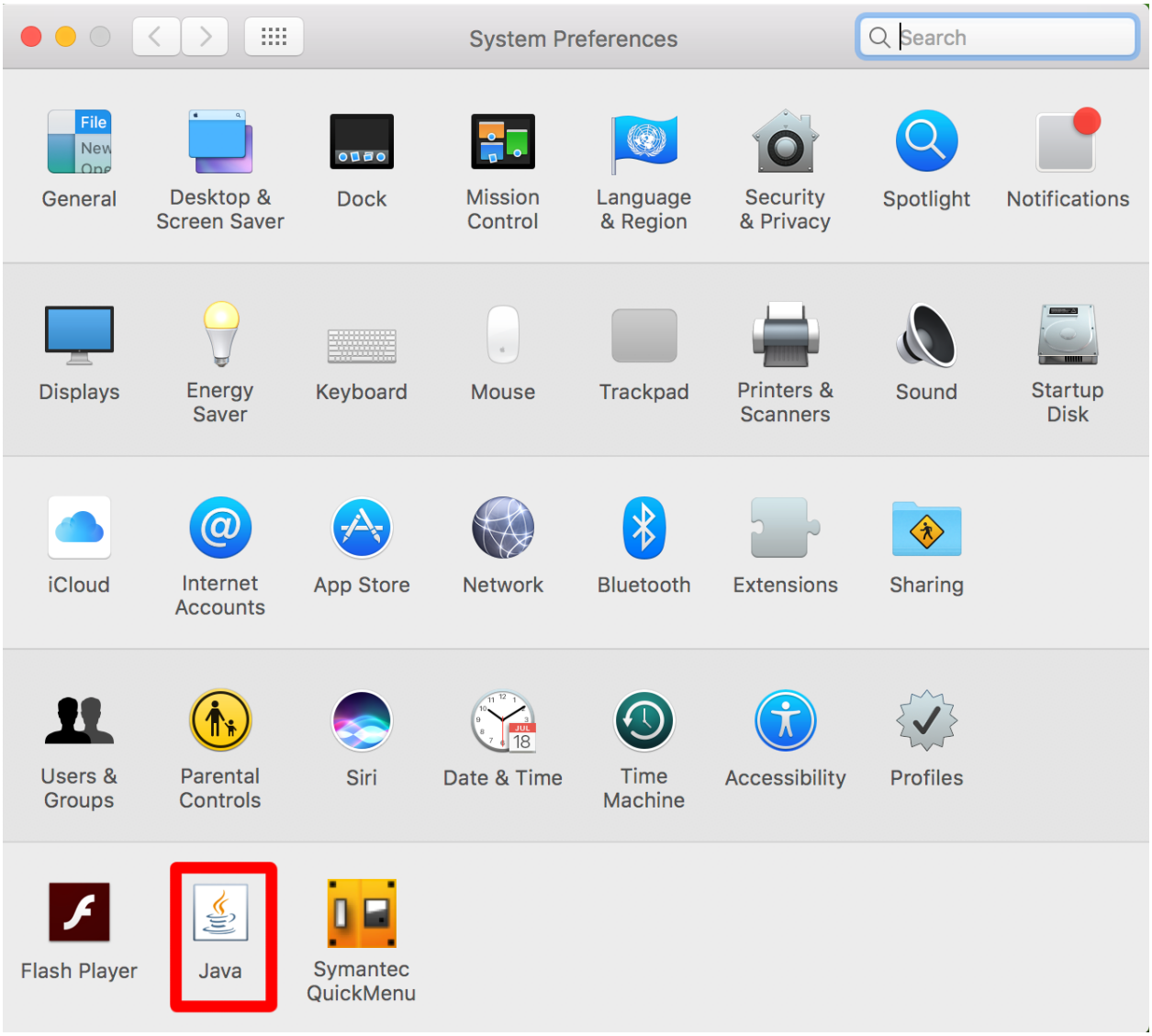
You should see *1.8* in the *Platform* column of the table.



1.2 Mac OSX

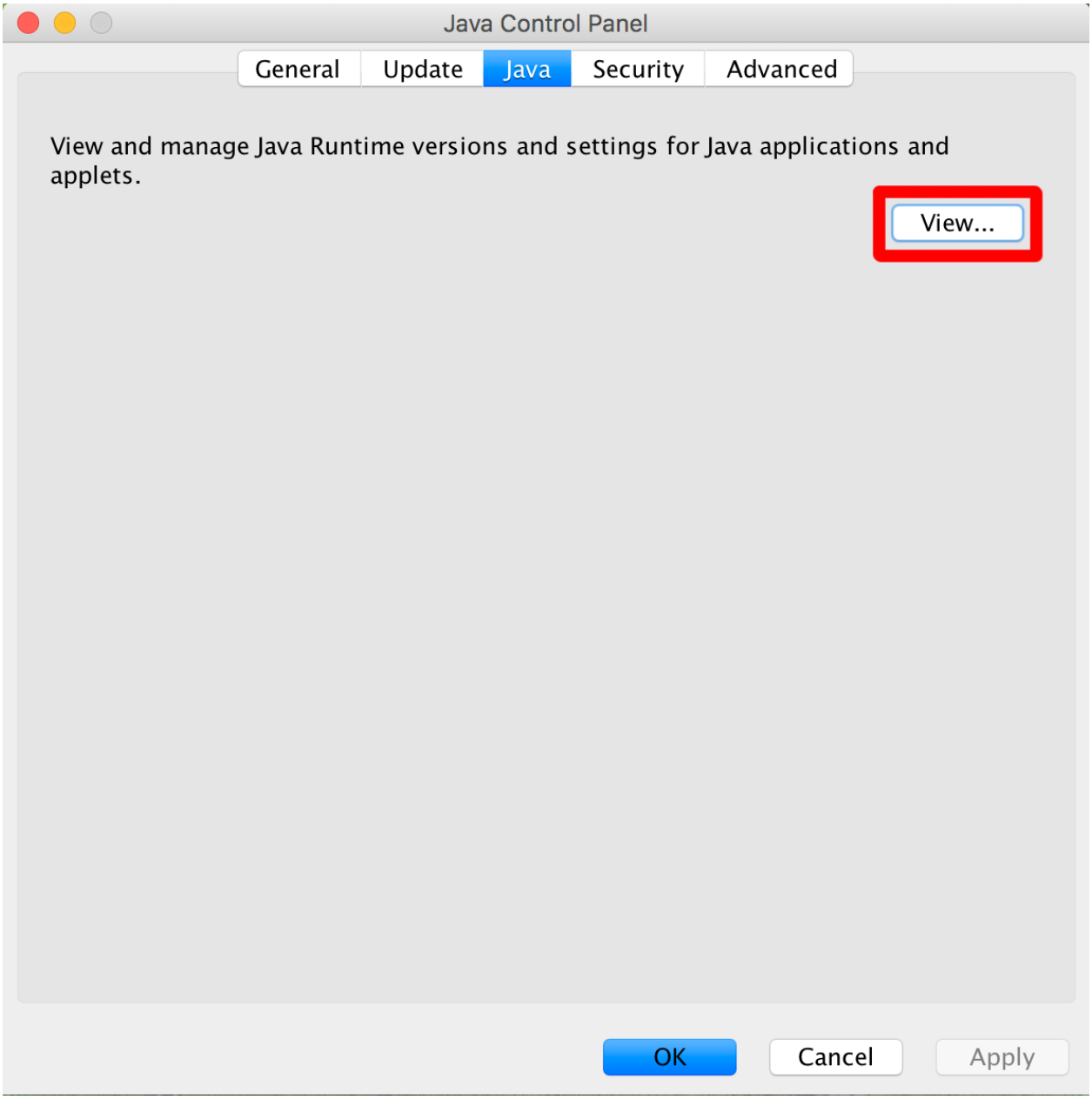
Mac users should follow these steps to figure out which version of Java is installed on the computer:

Open the **System preferences** and click on Java.

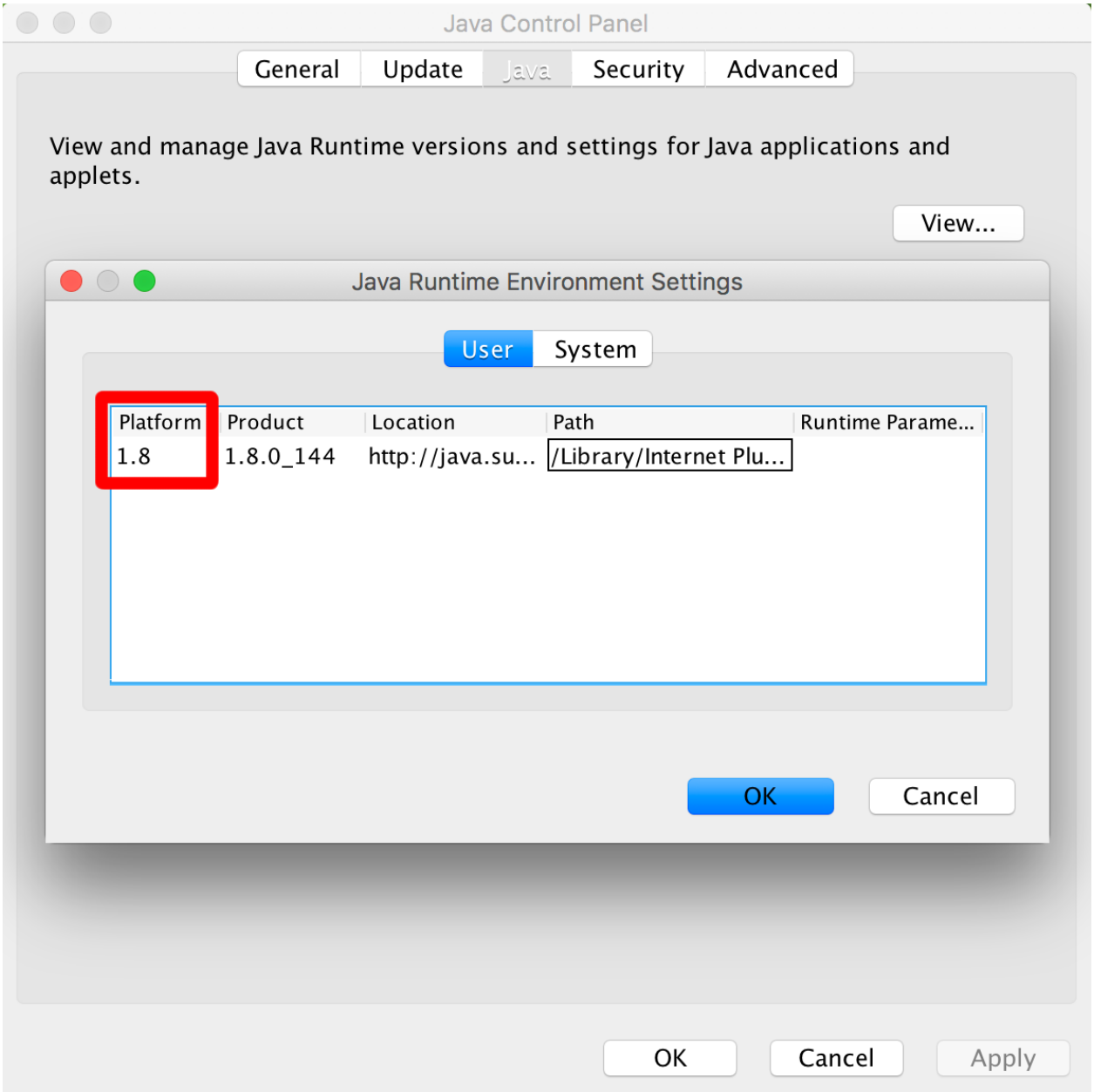


Select the **Java** tab at the top of the window (perhaps you might also want to install updates :D).

Click on the **View** button.



You should see *1.8* in the *Platform* column of the table.



1.3 Linux

You can determine what version of **Java** you have on your computer by entering the following command into the Terminal:

```
$ java -version
java version "1.8.0_161"
Java(TM) SE Runtime Environment (build 1.8.0_161-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.161-b12, mixed mode)
```

CHAPTER 2

Getting the app for your operating system & installation

2.1 Getting the app

2.1.1 Prebuilt app

Most users (Windows, Mac, Linux) should download the prebuilt `jar` file (executable app) available at HpoCaseAnnotator releases. To run HpoCaseAnnotator, you need to have Java Runtime Environment (Java version 8) installed on the machine. To run the app, a double click should work. Alternatively, enter following at the command line.

```
$ java -jar HpoCaseAnnotator.jar
```

2.1.2 Build from sources

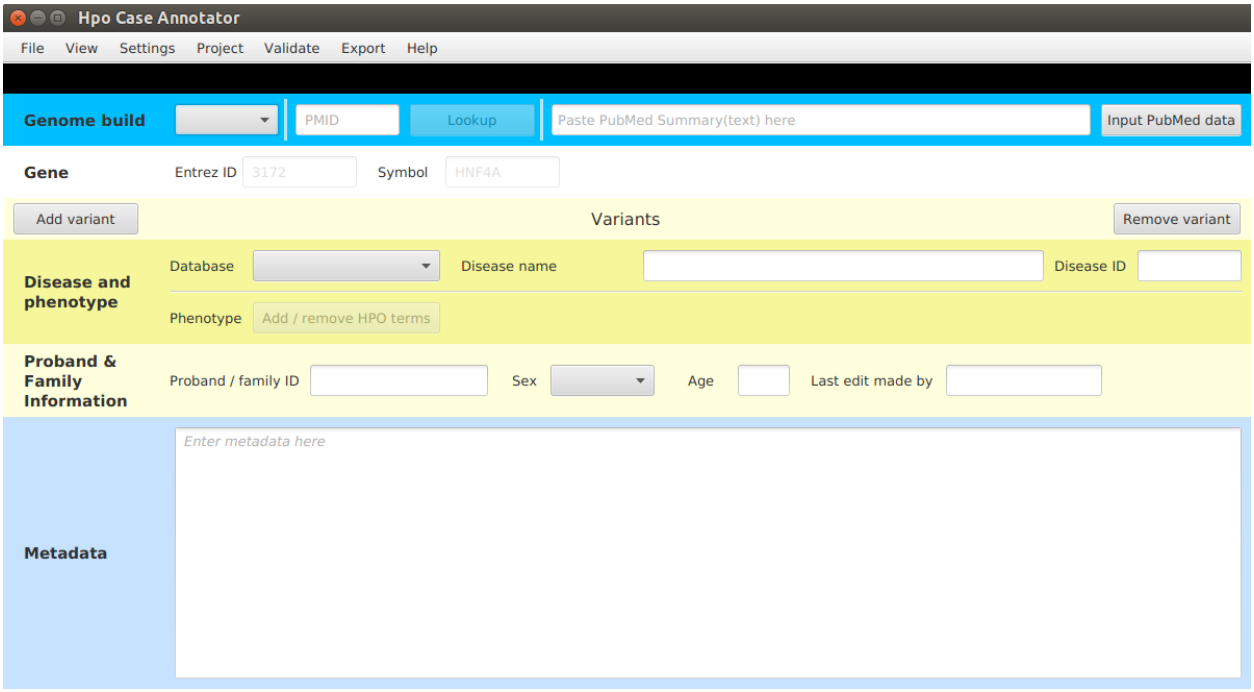
It is also possible to build the *HpoCaseAnnotator* from source code using Maven:

```
$ git clone https://github.com/monarch-initiative/HpoCaseAnnotator.git
$ cd HpoCaseAnnotator
$ mvn package
```

After successful build process the `HpoCaseAnnotator.jar` will be present in `hpo-case-annotator-gui/target` directory.

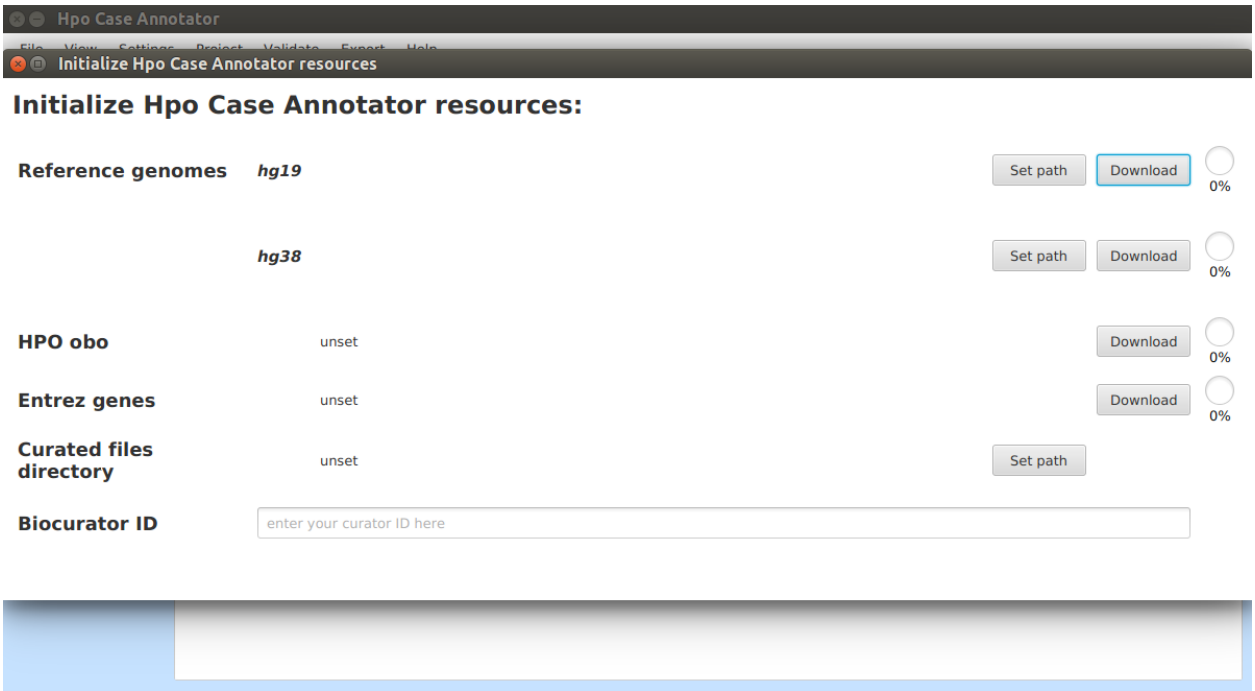
2.1.3 Initial setup

After a successful startup the dialog window will be opened:



Note, that *not* all the functionality is enabled after the first startup, since there are some resources that need to be downloaded first. Click on `Settings | Set resources` to start setting up the app.

A new dialog window will be opened:



2.1.4 Reference genome

Hpo Case Annotator needs access to the sequence of the reference genome in order to e.g. check whether the wild-type sequence entered for each variant matches the corresponding genomic position. *Hpo Case Annotator* is able to

download and pre-process the reference genome or you can provide the FASTA file yourself.

Currently, **GRCh37 (hg19)** and **GRCh38 (hg38)** genome assemblies are supported. For our internal use, we are still at hg19, and will use a liftover to move to hg38 coordinates!

- **Download and pre-process the reference genome sequences automatically**

In order to download the reference genome click on the *Download* buttons of individual genome assemblies. Each assembly file has roughly 1 GB and the download process may take up to 20 minutes depending on the speed of your internet connection. Do not close before the download & pre-processing is completed. After successful download, sequences of all the chromosomes will be concatenated into a single FASTA file and index will be created automatically using `HTS-JDK` library.

- **Provide FASTA file**

You do not have to download the reference genome files if there are already some present in your system. Use **Set path** buttons to set paths to local FASTA files. Corresponding index (*.fai) files should be present in the same directory.

This is all we need for the genomic position Q/C routines.

HPO obo

The app will automatically download the newest version of *Human Phenotype Ontology* (HPO) in `OBO` format, the file has ~5 MB. This needs to be done once (and can be updated as necessary). We download a copy of the OBO file so that the App can autocomplete the HPO terms names and labels.

Entrez genes

The app will also download information regarding genes. This is also a one-time operation and after the download autocompletion of gene symbols and IDs will be available.

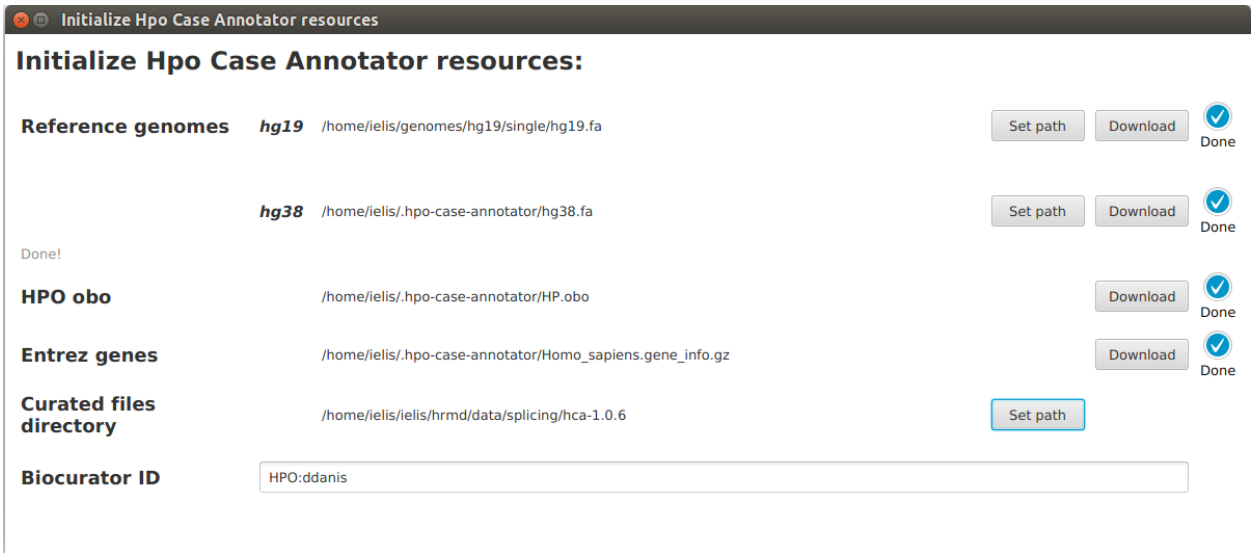
Curated files directory

Each curated case is stored as a file in `JSON` format. Here we set path to a directory where the JSON files created in a single project are stored.

Biocurator ID

Here provide your biocurator ID.

All of the resources are required to be downloaded just once and after these steps the app is fully prepared for work.


Initialize Hpo Case Annotator resources
Initialize Hpo Case Annotator resources:
Reference genomes
hg19
/home/ielis/genomes/hg19/single/hg19.fa
Set path
Download
Done
hg38
/home/ielis/.hpo-case-annotator/hg38.fa
Set path
Download
Done
Done!
HPO obo
/home/ielis/.hpo-case-annotator/HP.obo
Download
Done
Entrez genes
/home/ielis/.hpo-case-annotator/Homo_sapiens.gene_info.gz
Download
Done
Curated files directory
/home/ielis/ielis/hrmd/data/splicing/hca-1.0.6
Set path
Biocurator ID
HPO:ddanis

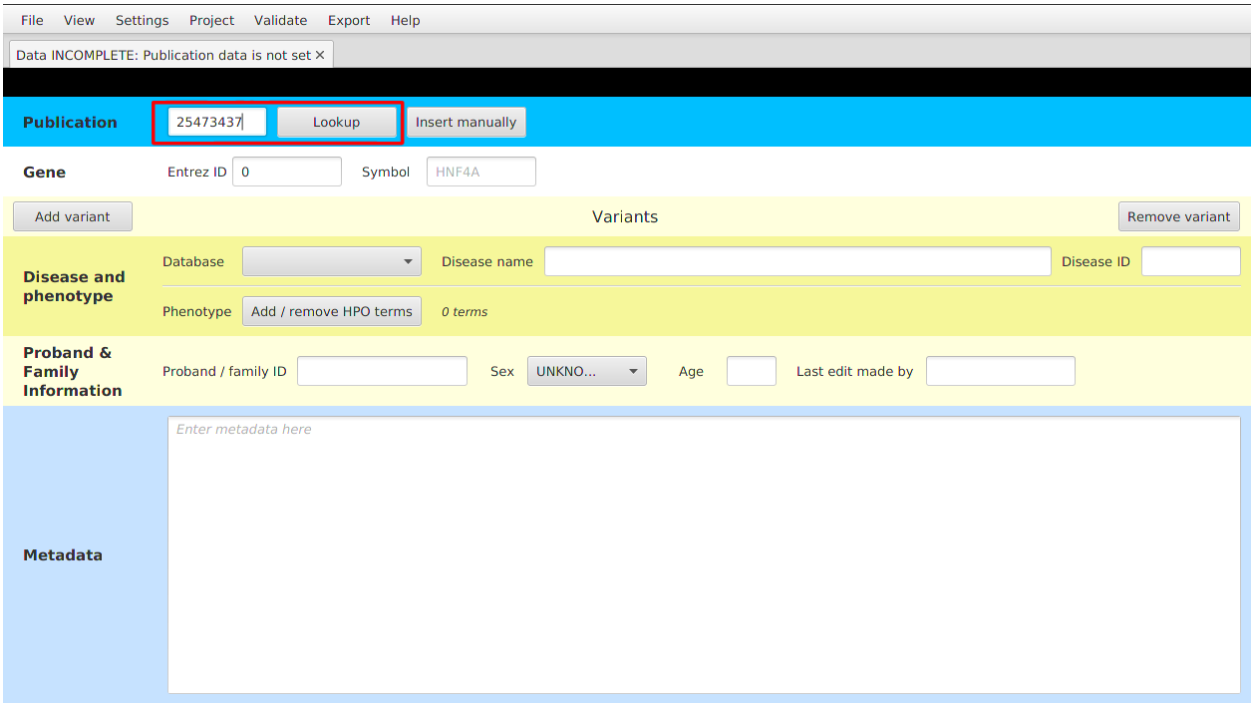
CHAPTER 3

Entering data

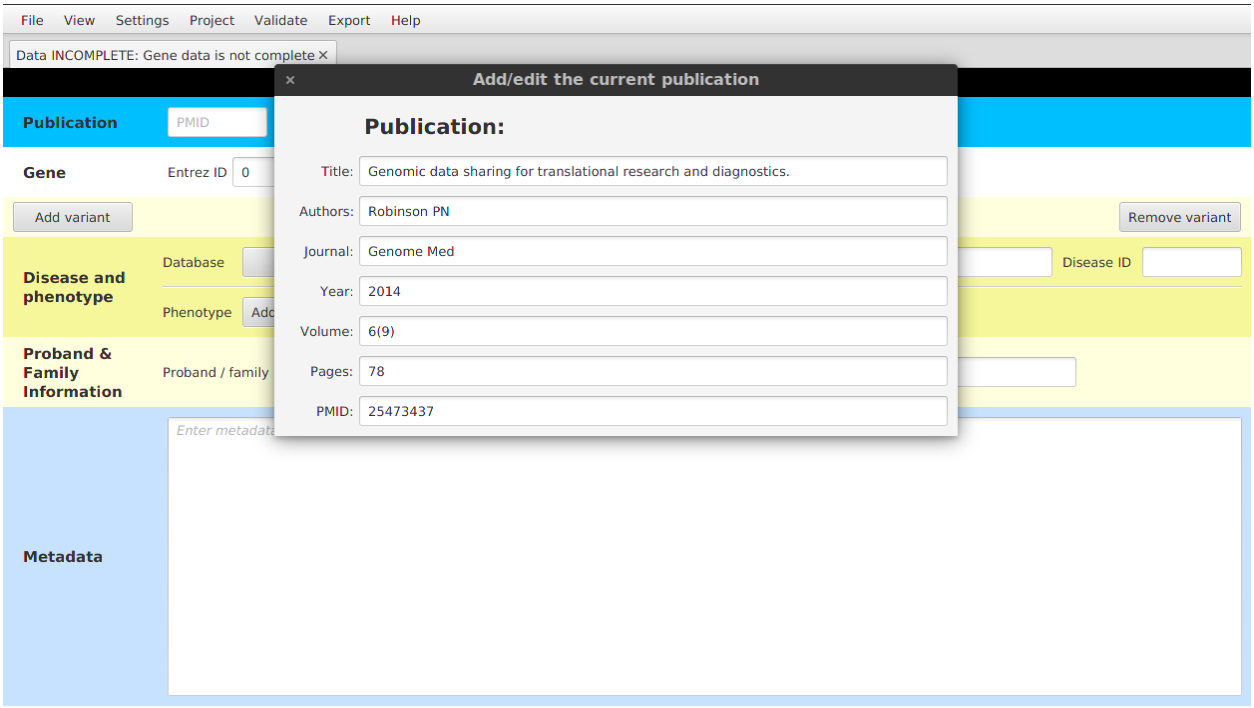
3.1 Publication

There are two ways of entering the data regarding the publication which describes the curated case:

1. *Using PMID* - enter the *PMID* number of the publication and hit the *Lookup* button. Publication details will be fetched from PubMed API, resulting in showing PMID and publication title.



2. *Entering the details manually* - click on the *Insert manually* button and enter all the details into the window that appears on the screen.



After setting the publication data, you can modify the data using `View | Show / edit current publication` menu item.

3.2 Genome build

For now, please use build 37 (called either "GRCh37" or "hg19"). Later, we will use the liftover utility of UCSC to add data for build 38.

3.3 Target Gene

In presumably almost all cases, we will know the target gene of the variant that has been published. We enter two bits of information:

- Entrez gene ID (e.g. 3172)
- gene symbol (e.g. *HNF4A*)

Note that the autocompletion is available for both fields, so usually entering just the gene symbol should be enough.

3.4 Variants

Click to *Add variant* button in order to create a new box for variant data. There are several variant types, where we store different set of variant validation metadata for each type.

3.4.1 Mendelian

Validation metadata important for the *Regulatory mendelian mutations (REMM)* project.

3.4.2 Somatic

Validation metadata for somatic variants.

3.4.3 Splicing

Data regarding splicing for the variants curated in the *Squirls* project.

3.4.4 Structural (Intrachromosomal/Translocation)

The way how we store data for variants stored in format denoted as *symbolic* in the VCF specs. These variants are usually longer (>100bp) deletions, duplications, inversions, etc.

We store the variants that affect a single chromosome using `INTRACHROMOSOMAL` variant type. The variants that affect multiple chromosomes (translocations/breakends) are stored as `TRANSCLOCATION` type.

3.4.5 Chromosome and position

Consult the article you are reading. I have found it helpful to see if the sequence surrounding the variant position is shown somewhere in the article. If this sequence is 20 nucleotides or more, you can use the BLAT tool of UCSC Genome Browser to find the corresponding position in the genome. If there are only a few bases, sometimes you can use guesswork to narrow things down enough to find the corresponding place in the genome. For older articles that specify the position of a variant using Genome Build 36 (called either "GRCH36" or "hg18"), you can use the UCSC Liftover utility. There are some articles that are of such low quality that it is simply not possible to reliably identify the chromosomal position of the variant. In these cases, the article should be rejected. It may also be worthwhile to consult dbSNP or ClinVar, since some published pathogenic variants are entered in these databases.

Note that position should be **one-based**, and *not* zero-based.

3.4.6 Reference / Alternative allele

For single-nucleotide variants, *Ref* and *Alt* are simply A,C,G, or T.

For deletions and insertions, please use the VCF format. Here is the Webpage with the latest details, but if in doubt please ask Peter. Just to give a simple example:

Let us pretend we have a ten base-pair reference sequence on chromosome `Z`:

```
ACGTAAGTCA
```

Let us imagine that the `T` at position 4 is deleted. This results in the sequence:

```
ACGAAGTCA
```

It might seem logical to write simple position=4, ref="T", alt="-". VCF format calls instead for this:

```
#CHROM POS ID REF ALT (other stuff)
Z 3 . GT G (other stuff)
```

This means that the dinucleotide at position 3-4 is affected and the variant sequence has only a G. For an insertion of a C between the T at position 4 and the A at position 5, we write:

```
#CHROM POS ID REF ALT (other stuff)
Z 4 . T TC . (other stuff)
```

We will use this convention, which will allow us to check the reference sequence and the position even for deletions, and should allow us a little more possibilities for Q/C-ing the genomic position etc.

3.4.7 Variant status

We need to enter information about whether the variant is **heterozygous** or **homozygous**. Note that if the patient has two different heterozygous mutations (i.e., is compound heterozygous), then we enter the second mutation in the second *Variant* box. In all other cases, we just use the first *Variant* box. Also, note that in some cases, the publications state (for an autosomal recessive disease) that *"the second mutation could not be found"*. Also in this case, do not enter anything into the second *Variant* box.

Note that if the first mutation is regulatory and the second mutation is coding (e.g., missense, nonsense, splicing, etc.), then you should use the category *coding* for the second mutation.

Finally, it is a good idea to use the Mutalyzer to check the nomenclature and location of the variants. The Mutalyzer will provide the surrounding genomic sequence for most variants, and this can be used to identify the genomic position of coding mutations using BLAT. It may also be useful to consult with ClinVar or the public version of HGMD about this.

3.4.8 Variant class

One of:

1. *promoter* - note that there are no really good definitions of where the promoter is located. Please put anything in the 5UTR in the class 5UTR, even if the effect seems to be on the promoter. Probably anything within 5-10,000 nucleotudes upstream of the transcription start site can be called promoter, but since we will have the numbers, we can do the classification automatically later. For now, I have taken the classification as mentioned in the original publications.
2. *enhancer* - regulatory region that is farther removed from the transcriptional start site than a promoter.
3. *5' UTR*
4. *3' UTR*
5. *microRNAgene* - here we mean any variation that affects the transcript that encodes for a microRNA (note: mutations that affect microRNA binding sites should in general be classified as *3' UTR*).
6. *RNP_RNA* - ribonucleoprotein (RNP) RNA component gene. These include ribosome and snRNP
7. *LINC_RNA* long intergenic non-coding RNA gene
8. *coding* - we only incldue coding mutations if the patient being described was compound heterozygous for a coding mutation and a regulatory mutation

Note that the *5' UTR* DNA sequences often form part of the actual promoter, and in general it is not possible to know if a variant affects the promoter function or the *5' UTR* function (which is of course in the mRNA and can affect the stability of the transcript). If a mutation is located in the *5' UTR*, then please enter *5' UTR* even if the effect is on the promoter. The data base and downstream analysis just has to know about this. In some cases, a mutation may be both *5' UTR* and promoter etc. Please enter the category that seems most relevant. We will automatically generate these annotations using `jannovar` anyway, so even variants with multiple categories will be correctly classified.

Note again that the category *coding* should only be used for the *second* mutation in compound heterozygous cases. At some point we may want to consider adding other classes, but none of the old data will be affected by a new class (e.g., silencer).

3.5 Disease data

Set the database (please use the OMIM id if at all possible). For OMIM, use the phenotype id, and not the gene id.

1. *Database*: one of OMIM or ORPHANET (use drop-down menu)
2. *Disease name*: please use a lower-case form of the canonical name, i.e., do not include all of the synonyms in upper-case letters.
3. *Database ID*: for OMIM; this will be a number like 614321

3.6 Phenotype data (HPO)

To enter or to modify the HPO data, you want and click on the *Add / remove HPO terms* button. Note that if you find you do not have enough, you can add additional terms with this button too.

A new window will be opened with *HPO tree browser* on the left side, *Text-mining analysis* on the right side and with table of *Approved terms* on the bottom-right side.

You should start typing name of the phenotypic trait into the text field above from the ontology tree. The text field has an autocompletion feature and helps you to identify the correct *HPO term label*. After completion of the label, click on the *Go* button to navigate to the term's position in the ontology tree.

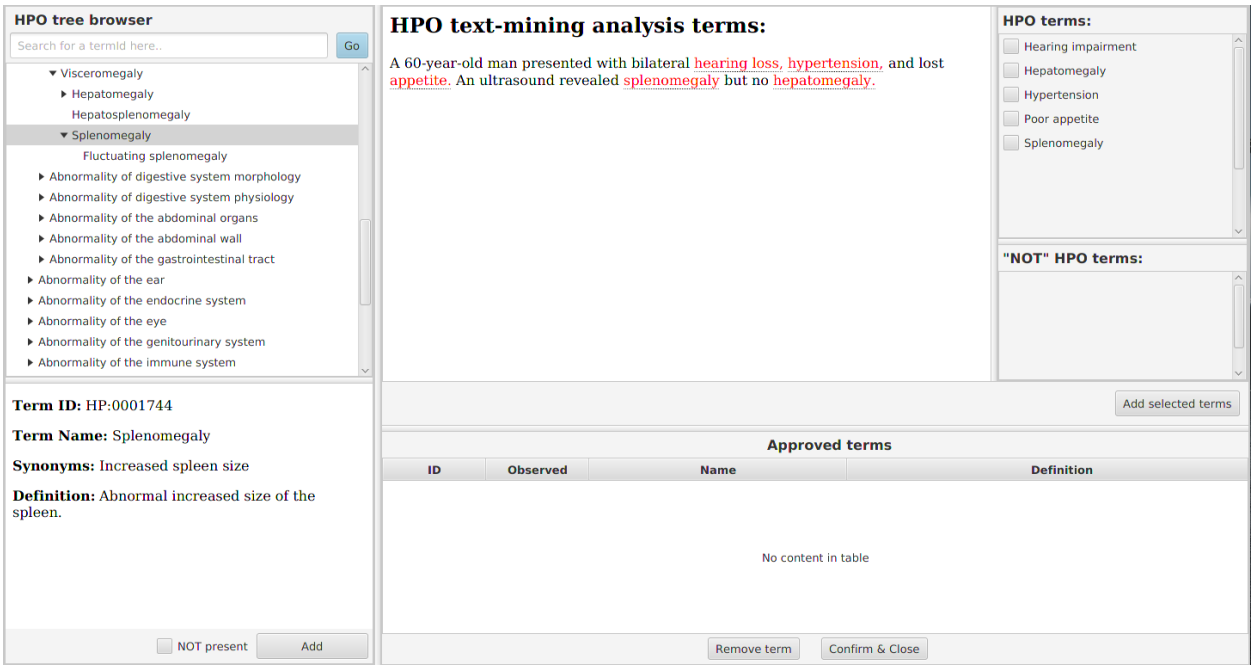
Then, you may want to look around the term in the ontology tree a bit and then approve the term's presence by hitting *Add* button at the bottom. The term will appear in the *Approved terms* table.

3.6.1 Text mining

In case you're curating variants from a publication that contains a clinical description of the proband's condition, *text mining* comes to help. To identify candidate HPO terms in a clinical description text, paste the text into the *Text-mining analysis* field.

Try the text-mining using e.g. the following toy example:

```
A 60-year-old man presented with bilateral hearing loss, hypertension, and lost␣
↪appetite.
An ultrasound revealed splenomegaly but no hepatomegaly.
```



Five HPO terms are picked up from the toy example. HPO term definition appears upon hovering with mouse upon the highlighted text. Clicking on the text will navigate you to the term definition within the ontology hierarchy (left panel). We recommend to read the text, approve the relevant terms on the right panel, and approving the mined terms by clicking on *Add selected terms* button.

Note: The previously used text-mining service was also able to identify *not* terms (e.g. no hepatomegaly). Unfortunately, the current service does not support this feature.

3.7 Proband & Family Information

The ID (patient/family identifier) is a free-text string that represents the ID used to designate the affected individual or family in the original paper. For instance, `family 3`. Note that we usually include all of the pathogenic variant in a given paper, but if little clinical data is given, and the phenotype is identical for two families, then it is OK to enter `family 3 and family 7`, say.

3.8 Metadata

Many of the individual papers about disease-causing variants have a lot of interesting additional information that is more or less heterogeneous. We would like to capture the most salient points in a free text that will be displayed on the planned website. For instance, here is an example `Metadata`:

```
The mutation is located in a 400-bp sequence located 25 kb downstream of PTF1A (the␣
↪gene
for pancreas-specific transcription factor 1a). This region acts as a developmental␣
↪enhancer
of PTF1A and that the mutations abolish enhancer activity. The mutation was shown to␣
↪abolish
binding of FOXA2 (Supplementary Figure 8 of Wheedon et al., 2014).
```

CHAPTER 4

Validation

For some of the uses of `HpoCaseAnnotator`, we enter not only the phenotype and genotype information, but also information about the molecular pathomechanism of the variant as well as any experimental methods that were used to validate the pathogenicity of the variant.

4.1 Non-coding variants

We have curated many non-coding variants that were used to validate the ‘‘Genomiser <https://www.ncbi.nlm.nih.gov/pubmed/27569544>‘_. As a rule, we only include a mutation if there is adequate evidence for its pathogenicity. As a general rule, there should be some experimental evidence for the mutation changing gene regulation of a target gene in some way. For some heavily studied genes, we will accept a mutation if it seems to be very similar to other published mutations (e.g., it lies on the same predicted transcription factor binding site as another mutation for which experimental evidence is available). Add as much evidence as possible. It is expected that at least one of the evidence categories will apply to each mutation.

1. reporter - Luciferase assay (or the similar CAT assay) to judge transcriptional activity. Indicate whether the mutation is associated with increased activity (up) or decreased activity (down) as compared with the wildtype construct (in percent).
2. EMSA - EMSA (electrophoretic mobility shift assay). This is used to indicate whether a protein binds to a given DNA sequence. For our purposes, we are referring to the protein affected by the mutation. Enter the corresponding protein if there is a change in binding. Enter the Entrez Gene ID and Gene Symbol of the protein that is affected by the mutation (usually a transcription factor)
3. cosegregation - enter yes if the mutation cosegregates with the disease in the family being investigated.
4. comparability - this is the weakest evidence class. Enter yes if the reason for believing that the mutation is pathogenic is simply that it is comparable to other published regulatory mutations in the gene.
5. other - this is for any other kind of experimental assay that shows an effect of a regulatory or non-coding mutation. Note that for now the categories are hard coded into the Java code, this should be put into some kind of configuration file in the future. The categories are at present:

Telomerase. Telomerase lengthening assay.

4.2 Splicing variants

For splicing variants, we include them if there is adequate evidence for missplicing and disease pathogenicity. TODO – describe.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search