Building the CellBase database

Sections in this page

This tutorial will first guide you to download a set of raw files from several data sources. These raw files shall contain the core data that will populate the Cellbase knowledgebase. Then, the tutorial will show you how to build the json documents that should be loaded into the Cellbase knowledgebase. Nevertheless, we have already processed all these data and json documents are available through our FTP server for those users who wish to skip these two sections below. Downloading raw files from the original sources and building the data models can be tricky. We encourage users to use our pre-built data models (*json* files) and to skip the *download* of raw files from original sources and the posterior *building* of the data models. Our pre-built json documents (data models) are available from

http://bioinfo.hpc.cam.ac.uk/downloads/cellbase/v4/homo_sapiens_grch37/mongodb/

http://bioinfo.hpc.cam.ac.uk/downloads/cellbase/v4/homo_sapiens_grch38/mongodb/

You could then directly jump to the Load data models section in this tutorial.

For those users willing to build CellBase knowledgbase from scratch, please follow the sections below.

Please note: allele population frequencies datasets are processed following a different pipeline and special sections can be found below for them.

Download Sources

Download can be done through the Cellbase CLI:

```
cellbase/build/bin$ ./cellbase.sh download

The following option is required: -d, --data

Usage:   cellbase.sh download [options]

Options:
      -a, --assembly       STRING     Name of
the assembly, if empty the first assembly in
configuration.json will be used
      --common             STRING     Directory
where common multi-species data will be
downloaded, this is mainly protein and
expression
                                        data
[<OUTPUT>/common]
      -C, --config         STRING     CellBase
configuration.json file. Have a look at

cellbase/cellbase-core/src/main/resources/confi
guration.json for an example
    * -d, --data           STRING     Comma
separated list of data to download: genome,
gene, variation, variation_functional_score,

regulation, protein, conservation,
clinical_variants, repeats, svs and 'all' to
download everything
      -h, --help                      Display
this help and exit [false]
      -L, --log-level      STRING     Set the
logging level, accepted values are: debug,
info, warn, error and fatal [info]
      -o, --output         STRING     The
output directory, species folder will be
created [/tmp]
      -s, --species        STRING     Name of
the species to be downloaded, valid format
include 'Homo sapiens' or 'hsapiens' [Homo
                                        sapiens]
      -v, --verbose        BOOLEAN
[Deprecated] Set the level of the logging
[false]
```

A number of datasets can be downloaded as indicated by the built-in documentation: genome, gene, variation, variation_functional_score, regulation, protein, conservation, clinical_variants, repeats, svs. An option `all` is implemented for the `--data` parameter to allow downloading all data by a single command. Some datasets (`genome` and `gene`) need the ENSEMBL perl API to be properly installed in order to be fully downloaded. Please note: all data can be downloaded, built and loaded in the database without the ENSEMBL API but some bits may be missing, e.g. gene xrefs.

For example, to download all human (GRCh37) data from all sources and save it into the `/tmp/data/cellbase/v4/` directory, run:

```
cellbase/build/bin$ ./cellbase.sh download -a GRCh37 --common
/tmp/data/cellbase/v4/common/ -d all -o /tmp/data/cellbase/v4/ -s
hsapiens
```

Please note: ensure you are located within the `cellbase/build/bin` directory before running the `download` command. Some perl scripts that use the ENSEMBL API may not be properly run otherwise. Also, note that COSMIC server requires login and therefore the CosmicMutantExport.tsv.gz file must be manually downloaded from their web page:

https://cancer.sanger.ac.uk/cosmic/download

Please, also note that heavy files will be downloaded and therefore the time needed for completion may vary between minutes and even hours. If download was successful, you can proceed to building the json objects that should be loaded into the corresponding database.

Downloading population frequencies datasets

Must be manually downloaded from source repositories:

- GONL: http://www.nlgenome.nl/
- gnomAD: https://data.broadinstitute.org/gnomAD
- 1000 Genomes Project: http://www.internationalgenome.org/
- UK10K: http://www.uk10k.org/data.html
- ESP: http://evs.gs.washington.edu/EVS/
- DiscovEHR: http://discovehrshare.com/downloads

ENSEMBL VCFs for ENSEMBL variation data must be manually downloaded as well:

ftp://ftp.ensembl.org/pub/release-90/variation/vcf/homo_sapiens/Homo_sapiens.vcf.gz

ftp://ftp.ensembl.org/pub/release-90/variation/vcf/homo_sapiens/Homo_sapiens_somatic.vcf.gz

Build Data Models

The process may be carried out by using the Cellbase CLI:

```
cellbase/build/bin$ ./cellbase.sh build
The following options are required: -d, --data
-i, --input

Usage:   cellbase.sh build [options]

Options:
      -a, --assembly         STRING     Name of
the assembly, if empty the first assembly in
configuration.json will be used
      --common               STRING
Directory where common multi-species data will
be downloaded, this is mainly protein and
expression
                                         data
[<OUTPUT>/common]
      -C, --config           STRING
CellBase configuration.json file. Have a look
at

cellbase/cellbase-core/src/main/resources/confi
guration.json for an example
```

```
    * -d, --data              STRING     Comma
separated list of data to build: genome,
genome_info, gene, variation,

variation_functional_score, regulation,
protein, ppi, conservation, drug,
clinical_variants,

repeats, svs. 'all' builds everything.
      --flexible-gtf-parsing            By
default, ENSEMBL GTF format is expected.
Nevertheless, GTF specification is quite loose
and
                                        other
GTFs may be provided in which the order of the
features is not as systematic as within the

ENSEMBL's GTFs. Use this option to enable a
more flexible parsing of the GTF if it does not
strictly
                                        follow
ENSEMBL's GTFs format. Flexible GTF requires
more memory and is less efficient. [false]
      -h, --help                        Display
this help and exit [false]
    * -i, --input             STRING     Input
directory with the downloaded data sources to
be loaded
      -L, --log-level         STRING     Set the
logging level, accepted values are: debug,
info, warn, error and fatal [info]
      -o, --output            STRING     Output
directory where the JSON data models are saved
[/tmp]
      -s, --species           STRING     Name of
the species to be built, valid format include
'Homo sapiens' or 'hsapiens' [Homo sapiens]
      -v, --verbose           BOOLEAN
[Deprecated] Set the level of the logging
[false]
```

The `build` process will integrate data from the different sources into the corresponding data models. Use the Cellbase CLI for building the data models. For example, build all human (GRCh37) data models reading the files from the `/tmp/data/cellbase/v4/homo_sapiens_grch37/` directory created in section Download Sources and save the result at `/tmp/data/cellbase/v4/homo_sapiens_grch37/mongodb/`:

```
cellbase/build/bin$ mkdir
/tmp/data/cellbase/v4/homo_sapiens_grch37/mongo
db
cellbase/build/bin$ ./cellbase.sh build -a
GRCh37 --common /tmp/data/cellbase/v4/common/
-d all -i
/tmp/data/cellbase/v4/homo_sapiens_grch37/ -o
/tmp/data/cellbase/v4/homo_sapiens_grch37/mongo
db/ -s hsapiens
```

Note: building process for the whole CellBase dataset may require up to 16GB of RAM and may take up to ~24h, depending on the hardware.

Building variation data models

First, allele population frequencies datasets must be processed. In order to do this, VCF files downloaded in "Downloading population frequencies datasets" Section must be loaded into an OpenCGA installation. Please, refer to Index Pipelines, Variant Storage Operations#Index and Getting Started in 5 minutes for further details on how to do this with an OpenCGA installation.

Once the OpenCGA installation is fully loaded, allele population frequencies must be parsed and/or calculated for each study. Please, refer to Variant Storage Operations#CalculateStatistics for further details on how to perform this operation with an OpenCGA installation.

Then, population frequencies must be exported into CellBase compliant *json* files. Please, refer to Variant Storage Operations#Exportfrequencies(statistics) for further details on how to perform this operation with an OpenCGA installation.

With this, population frequencies are ready to be processed by CellBase. However, before being able to run the CellBase CLI that will generate the final *.json.gz* files, ENSEMBL variation's VCF file must be split into multiple files: one per chromosome:

```
cellbase/build/bin$ for chr in {1..22} X Y MT;
do echo zgrep "^#\|^$chr"
/tmp/data/cellbase/v4/homo_sapiens_grch37/varia
tion/Homo_sapiens.vcf.gz | gzip -c >
/tmp/data/cellbase/v4/homo_sapiens_grch37/varia
tion/Homo_sapiens."$chr".vcf.gz; done
```

All data is now ready for generating the final *variation json* files.

Please note, actual *build* of the variation collection is currently not performed by using the *build* command in the CLI, but by annotating ENSEMBL variation VCFs using the *variant-annotation* command line. Thus, running the following command line for each VCF downloaded in previous step will generate the *variation* files (below showing one run for chromosome 1):

```
cellbase/build/bin$ cellbase/bin/cellbase.sh
variant-annotation -a GRCh37 -s hsapiens
--exclude
variation,populationFrequencies,expression,gene
Disease,drugInteraction -i
/tmp/data/cellbase/v4/homo_sapiens_grch37/varia
tion/Homo_sapiens.1.vcf.gz -o
/tmp/data/cellbase/v4/homo_sapiens_grch37/mongo
db/variation_chr1.json.gz
-Dpopulation-frequencies=/tmp/data/frequencies/
chr1.freq.cellbase.json.gz
```

After completion of the build process, your output directory shall look like:

```
cellbase/build/bin$ ls
/tmp/data/cellbase/v4/homo_sapiens_grch37/mongo
db/
clinical_variants.full.json.gz
clinvarVersion.json
conservation_10.json.gz
conservation_11.json.gz
conservation_12.json.gz
conservation_13.json.gz
conservation_14.json.gz
conservation_15.json.gz
conservation_16.json.gz
conservation_17.json.gz
conservation_18.json.gz
conservation_19.json.gz
conservation_1.json.gz
conservation_20.json.gz
conservation_21.json.gz
conservation_22.json.gz
conservation_2.json.gz
conservation_3.json.gz
conservation_4.json.gz
conservation_5.json.gz
conservation_6.json.gz
conservation_7.json.gz
conservation_8.json.gz
conservation_9.json.gz
conservation_M.json.gz
conservation_X.json.gz
conservation_Y.json.gz
cosmic.json.gz
cosmicVersion.json
dgidbVersion.json
dgvVersion.json
disgenetVersion.json
ensemblCoreVersion.json
ensemblRegulationVersion.json
```

```
ensemblVariationVersion.json
geneExpressionAtlasVersion.json
gene.json.gz
genome_info.json
genome_info.log
genome_sequence.json.gz
genomeVersion.json
genomicSuperDups.json
gnomadVersion.json
hpoVersion.json
interproVersion.json
mirbaseVersion.json
phastConsVersion.json
phyloPVersion.json
protein.json.gz
prot_func_pred_chr_10.json
prot_func_pred_chr_11.json
prot_func_pred_chr_12.json
prot_func_pred_chr_13.json
prot_func_pred_chr_14.json
prot_func_pred_chr_15.json
prot_func_pred_chr_16.json
prot_func_pred_chr_17.json
prot_func_pred_chr_18.json
prot_func_pred_chr_19.json
prot_func_pred_chr_1.json
prot_func_pred_chr_20.json
prot_func_pred_chr_21.json
prot_func_pred_chr_22.json
prot_func_pred_chr_2.json
prot_func_pred_chr_3.json
prot_func_pred_chr_4.json
prot_func_pred_chr_5.json
prot_func_pred_chr_6.json
prot_func_pred_chr_7.json
prot_func_pred_chr_8.json
prot_func_pred_chr_9.json
prot_func_pred_chr_MT.json
prot_func_pred_chr_X.json
prot_func_pred_chr_Y.json
regulatory_region.json.gz
repeats.json.gz
simpleRepeat.json
structuralVariants.json.gz
toload
uniprotVersion.json
uniprotXrefVersion.json
variation_chr10.json.gz
variation_chr10.somatic.json.gz
variation_chr11.json.gz
variation_chr11.somatic.json.gz
variation_chr12.json.gz
variation_chr12.somatic.json.gz
```

```
variation_chr13.json.gz
variation_chr13.somatic.json.gz
variation_chr14.json.gz
variation_chr14.somatic.json.gz
variation_chr15.json.gz
variation_chr15.somatic.json.gz
variation_chr16.json.gz
variation_chr16.somatic.json.gz
variation_chr17.json.gz
variation_chr17.somatic.json.gz
variation_chr18.json.gz
variation_chr18.somatic.json.gz
variation_chr19.json.gz
variation_chr19.somatic.json.gz
variation_chr1.json.gz
variation_chr1.somatic.json.gz
variation_chr20.json.gz
variation_chr20.somatic.json.gz
variation_chr21.json.gz
variation_chr21.somatic.json.gz
variation_chr22.json.gz
variation_chr22.somatic.json.gz
variation_chr2.json.gz
variation_chr2.somatic.json.gz
variation_chr3.json.gz
variation_chr3.somatic.json.gz
variation_chr4.json.gz
variation_chr4.somatic.json.gz
variation_chr5.json.gz
variation_chr5.somatic.json.gz
variation_chr6.json.gz
variation_chr6.somatic.json.gz
variation_chr7.json.gz
variation_chr7.somatic.json.gz
variation_chr8.json.gz
variation_chr8.somatic.json.gz
variation_chr9.json.gz
variation_chr9.somatic.json.gz
variation_chrMT.json.gz
variation_chrX.json.gz
```

```
variation_chrX.somatic.json.gz
variation_chrY.json.gz
windowMasker.json
```

If `build` was successful, you can proceed to loading the data models into the database.

Load Data Models

Please, note that before loading the data models into the database CellBase *configuration.json* must have been appropriately configured indicating the database host names, ports, user and password.

CellBase code is open-source and freely available at **https://github.com/opencb/cellbase**

Use the CellBase CLI to load the data models:

```
For example, to load all human (GRCh37) data models from the /tmp/data/ce
llbase/v4/homo_sapiens_grch37/mongodb/ created in section "Build Data
Models", into the cellbase_hsapiens_grch37_v4 database and creating the
indexes as indicated in the .js scripts within cellbase/cellbase-app/app/
mongodb-scripts/, run:

cellbase/build/bin$ ./cellbase.sh load -d variation --database
cellbase_hsapiens_grch37_v4 -i
/mnt/data/downloads/cellbase/v4/homo_sapiens_grch37/mongodb/ -L debug
-Dmongodb-index-folder=/home/cafetero/appl/dev/cellbase/cellbase-app/app/
mongodb-scripts/
```

Please, note that the whole loading and indexing process may need ~24h to complete, depending on the available hardware.

After successful load of all data, the corresponding database shall look like:

```
$ mongo mongodb-dev/cellbase_hsapiens_grch37_v4
MongoDB shell version: 3.0.9
connecting to:
mongodb-dev/cellbase_hsapiens_grch37_v4clinical
_variants
> show collections;
clinical_variants
conservation
gene
genome_info
genome_sequence
metadata
protein
protein_functional_prediction
protein_protein_interaction
regulatory_region
repeats
variation
variation_functional_score
```