# MakeHub User Guide

## Author and Contact Information

Katharina J. Hoff

University of Greifswald,
Institute for Mathematics and Computer Science,
Walther-Rathenau-Str. 47,
17489 Greifswald

University of Greifswald,
Center for Functional Genomics of Microbes,
Felix-Hausdorff-Str. 8,
17489 Greifswald

katharina.hoff@uni-greifswald.de

# Contents

# What is MakeHub?

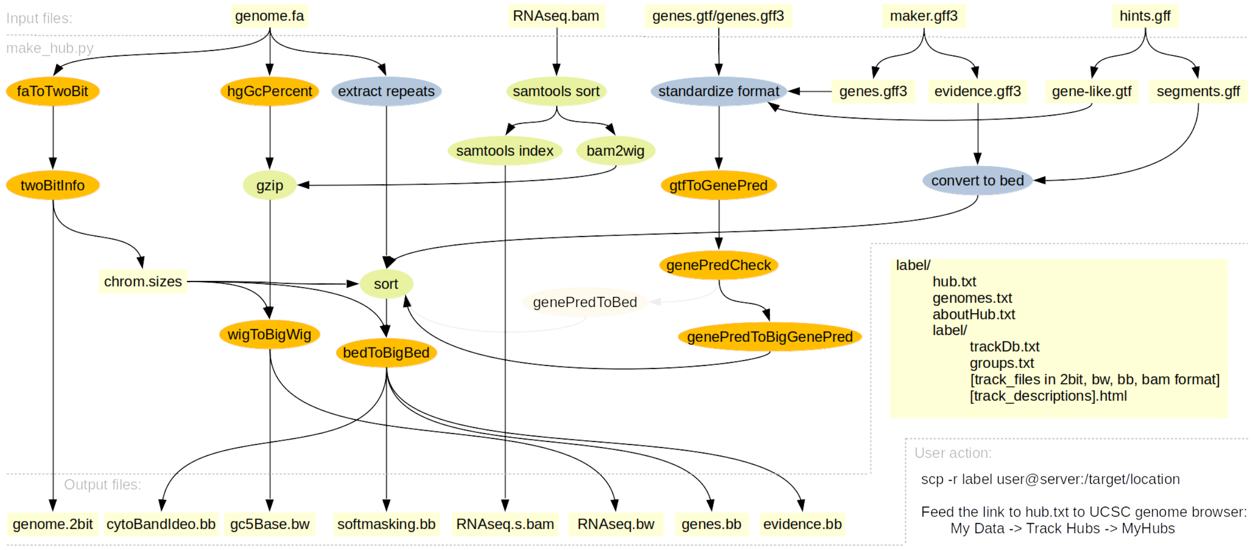
MakeHub is a command line tool for the fully automatic generation of of track data hubs[1] for visualizing genomes with the UCSC genome browser[2]. Track data hubs are data structures that contain all required information about a genome for visualizing with the UCSC genome browser.

Assembly hubs need to be hosted on a publicly available webspace (that might be user/password protected) for usage with the UCSC genome browser.

MakeHub is implemented in Python3 and automatically executes tools provided by UCSC for generation of assembly hubs (http://hgdownload.soe.ucsc.edu/admin/exe) on Linux and MacOS X x86_64 computers. For visualization of RNA-Seq alignment data from BAM files, MakeHub uses Samtools[3]]. If installed, the AUGUSTUS[4] tool bam2wig is used to speed up BAM to wig format conversion (https://github.com/Gaius-Augustus/Augustus), which is otherwise performed without bam2wig.

MakeHub can either be used to create entirely new assembly hubs, or it can be used to add tracks to hubs that were previously created by MakeHub.

For display by the UCSC Genome Browser, assembly hubs need to be hosted on a publicly accessible web server.



# Installation

## Quick Start

MakeHub is a Python3 script for Linux or MacOS X with x86-64 architecture. It requires Python3, Biopython, gzip, sort and - in the case that BAM files are provided - samtools, and optionally the AUGUSTUS tool bam2wig.

Many users who create the input data for MakeHub, e.g. with BRAKER [5], have the required dependencies already installed on their system and may thus skip ahead to section Running MakeHub. In case of doubt, read the following sections about installation of Dependencies and MakeHub installation.

## Dependencies

In the following, we give instructions on where dependencies can be obtained, and how they may be installed on Ubuntu Linux.

Python3 is available from https://www.python.org/downloads/, or as package for many Unix systems. Choose version 3.5 or newer (because otherwise, subprocess module is not fully functional).

For example, on Ubuntu, install Python3 with:

```
sudo apt install python3
```

We recommend to use pip for installing further python modules. pip is available at https://pypi.org/project/pip/. It is also available as package for many Unix systems.

For example, on ubuntu, install pip with:

```
sudo apt install python3-pip
```

Further, MakeHub uses Biopython (e.g. for parsing a genome file in order to determine which parts of the genome have been masked for repeats). Install biopython with pip as follows:

```
pip3 install biopython
```

MakeHub uses the following tools provided by UCSC at http://hgdownload.soe.ucsc.edu/admin/exe:

- bedToBigBed
- genePredCheck
- faToTwoBit
- gtfToGenePred
- hgGcPercent
- ixIxx
- twoBitInfo
- wigToBigWig
- genePredToBed
- genePredToBigGenePred (optional)

You may download these binaries and make them available in your $PATH. However, if you skip installing these tools, they will be downloaded during MakeHub execution, automatically.

In rare cases, particularly on older x86_64 Unix systems, the UCSC tools might throw errors because they are not statically linked in all parts, i.e. they will try to use some old system libraries and crash. If you observe this, try downloading the sources of KentUtils from github. We have had the best experience with compiling Kent tools for MakeHub from https://github.com/ENCODE-DCC/kentUtils/.

MakeHub uses two `*as` -files from UCSC:

- http://genome.ucsc.edu/goldenPath/help/examples/bigGenePred.as
- http://genome.ucsc.edu/goldenPath/help/examples/cytoBand.as

MakeHub will automatically download both files if they are not already present in the directory where `make_hub.py` or UCSC tools reside.

MakeHub uses Samtools for BAM file sorting and conversion. Samtools is avilable at https://github.com/samtools/. It is also avilable as package with many linux distributions.

For example, on ubuntu, install samtools with:

```
sudo apt install samtools
```

MakeHub has been tested with Samtools 1.8-20-g4ff8062. It is not fully downward compatible with older versions (we have for example tried samtools 1.1 and that is incompatible). How to know whether your samtools version is compatible? The samtools calls by `make_hub.py` are of the following syntax:

```
samtools sort -@ INT file.bam -o out.bam
samtools index -@ INT file.bam file.bam.bai
samtools mpileup -o file.pu file.bam
```

At some point in time, the samtools usage changed so that the output option `-o out.bam` became possible. If you type `samtools sort --help`, you want to find a line that says

`-o FILE Write final output to FILE rather than standard output` then your samtools are most likely compatible.

MakeHub uses gzip for compressing wig files that were created from BAM files. gzip is available at https://ftp.gnu.org/gnu/gzip/. It often installed by default on Unix systems. If not, it is usually available as a package.

If missing, on Ubuntu, install with:

```
sudo apt install gzip
```

MakeHub uses Unix sort. sort should be installed by default on all Unix systems.

MakeHub can use the AUGUSTUS tool bam2wig, if that tool is available in the $PATH. bam2wig is available as part of AUGUSTUS at https://github.com/Gaius-Augustus/Augustus. Please follow the compilation instructions in Augustus/auxprogs/bam2wig/README.txt in case the default make command fails.

## MakeHub

MakeHub is a python3 script named make_hub.py. It does not require a particular installation procedure after download.

It can be executed either with

```
python3 make_hub.py
```

If you add make_hub.py to your $PATH (i.e. by adding the location of make_hub.py at the bottom of your ~/.bashrc file similar to `PATH=/path/to/MakeHub:$PATH`, followed by loading the ~/.bashrc file in case you did not re-open a new bash session with `source ~/.bashrc`)and make it executable (i.e. with `chmod u+x make_hub.py`), it can be executed with

```
make_hub.py
```

from any location on your computer.

# Data Preparation

MakeHub accepts files in the following formats:

- genome file in FASTA format (simple FASTA headers without whitespaces or special characters); if the file is softmasked, a track with repeat information will automatically be generated. Note that the FASTA headers must be consistent with BAM-, hints- and gene prediction files.
- BAM file(s) with RNA-Seq to genome alignments
- gene prediction file(s) in GTF-format, e.g. from BRAKER
- AUGUSTUS hints files in BRAKER-specific GFF hints format
- Gene prediction files in GFF3-format from MAKER [6], Gemoma [7], SNAP [8]and GlimmerHMM [9].

# Running MakeHub

MakeHub can be used either to create new assembly hubs, or to add tracks to assembly hubs that had previously been created.

## Creating a new hub

The essential arguments for creating a new assembly hub are:

- `-e EMAIL` , `--email EMAIL`
  Contact e-mail adress for assembly hub. This e-mail adress will be displayed on all HTML pages that describe this hub and its tracks. Providing an e-mail adress is a requirement for UCSC assembly hubs, e.g. described at http://genomewiki.ucsc.edu/index.php/Assembly_Hubs and http://genomewiki.ucsc.edu/index.php/Public_Hub_Guidelines#Track_description_page_recommendations.

- `-g GENOME` , `--genome GENOME`
  Genome file in FASTA format. If the file contains softmasked repeats, a repeat masking track with softmasking information will automatically be generated.

- `-l SHORT_LABEL` , `--short_label SHORT_LABEL`
  Short label (without whitespaces and special characters) for identifying assembly hub, will also be used as directory name for hub, e.g. `--short_label fly` . Be aware that our understanding of a 'short label' is slightly different from the understanding of the UCSC Genome Browser group, who desire a slightly more descriptive short label than make_hub.py (see section Steps to take prior adding a MakeHub Track Data Hub to the UCSC public hub list).

At the point in time of assembly hub creation, we strongly recommend the additional usage of

- `-L LONG_LABEL` , `--long_label LONG_LABEL`
  Long label for hub, e.g. english organism name, if it contains whitespaces, pass it with quotation marks: `---long_label "fruit fly"`

You may at the point of time of creating a hub already supply information about all gene prediction and evidence tracks that you would like to see in your final hub. Please have a look at the section Options Explained for information about possible tracks. The section also describes how to add latin species name and assembly version.

Usage example 1:

```
make_hub.py -l hmi1 -L "Rodent tapeworm" -g data/genome.fa -e \
  katharina.hoff@uni-greifswald.de
```

The resulting hub is trivial, as it only displays very basic information about the genome, such as the GC-content, restriction enzyme sites and repeat masking segments.

If you want to visualize the result, connect the following hub with the UCSC genome browser (see section How to use MakeHub output with UCSC Genome Browser): http://augustus.uni-greifswald.de/bioinf/makehub/examples/hmi1/hub.txt

Usage example 2:

```
make_hub.py -l hmi2 -L "Rodent tapeworm" -g data/genome.fa -e \
  katharina.hoff@uni-greifswald.de -a data/annot.gtf -b data/rnaseq.bam \
  -d
```

In comparison to the first example, the resulting hub has a track with reference annotation genes, and a track with coverage information from RNA-Seq data, and it displays the native BAM-file ( `-d` ).

If you want to visualize the result, connect the following hub with the UCSC genome browser (see section How to use MakeHub output with UCSC Genome Browser): http://augustus.uni-greifswald.de/bioinf/makehub/examples/hmi2/hub.txt

Usage example 4:

```
make_hub.py -l hmi4 -L "Rodent tapeworm" -g data/genome.fa -e \
  katharina.hoff@uni-greifswald.de -a data/annot.gtf -b data/rnaseq.bam \
  -d -X data -M data/maker.gff -E data/gemoma.gff \
  -I data/glimmer.gff -S data/snap.gff \
  -N "Hymenolepsis microstoma" -V GCA_000469805.2
```

In comparison to the first two examples, the resulting hub has a large number of evidence and gene prediction tracks from BRAKER, MAKER, Gemoma, GlimmerHMM and SNAP.

If you want to visualize the result, connect the following hub with the UCSC genome browser (see section How to use MakeHub output with UCSC Genome Browser): http://augustus.uni-greifswald.de/bioinf/makehub/examples/hmi4/hub.txt

## Adding tracks to existing hub

If a hub already exists, you may add tracks to this existing hub using the option `-A`, `--add_track`. The minimal required arguments (besides giving the approriate information that you would like to add) are:

- `-A`, `--add_track`
  Add track(s) to existing hub

- `-e EMAIL`, `--email EMAIL`
  Contact e-mail adress for assembly
  hub.

- `-l SHORT_LABEL`, `--short_label SHORT_LABEL`
  Short label (without whitespaces and special characters) for identifying assembly hub.

Usage example 3:

First, we create a novel track hub hmi3 that is similar to Usage example 2:

```
make_hub.py -l hmi3 -L "Rodent tapeworm" -g data/genome.fa -e \
  katharina.hoff@uni-greifswald.de -a data/annot.gtf -b data/rnaseq.bam \
  -d
```

Subsequently, we add a number of tracks:

```
make_hub.py -l hmi3 -e katharina.hoff@uni-greifswald.de -i data/hintsfile.gff \
  -A -M data/maker.gff -X data
```

The resulting hub has many gene prediction tracks from the BRAKER output directory `data`, and from the MAKER output file `data/maker.gff`.

Let's add one more track (only for the sake of demonstration, this track could have been included in the previous example, of course, or at the point of time of track generation):

```
make_hub.py -l hmi3 -e katharina.hoff@uni-greifswald.de -i data/hintsfile.gff \
  -A -E data/gemoma.gff
```

If you want to visualize the result, connect the following hub with the UCSC genome browser (see section How to use MakeHub output with UCSC Genome Browser): http://augustus.uni-greifswald.de/bioinf/makehub/examples/hmi3/hub.txt

## Options explained

In the following, we explain all options of make_hub.py

- `-h, --help`
  Print help message and exit.

- `-p, --printUsageExamples`
  Print usage examples for make_hub.py to command line (for demonstration).

- `-e EMAIL, --email EMAIL`
  Contact e-mail adress for assembly hub. This is a requirement for all publicly listed assembly hubs. It is obligatory for make_hub.py.

- `-g GENOME, --genome GENOME`
  Genome file in FASTA format. If the file is softmasked for repeats, a repeat masking track will automatically be generated, unless the option:

- `-n, --no_repeats`
  Disable repeat track generation from softmasked genome sequence is activated (this may save runtime, particularly for large genomes).

- `-L LONG_LABEL, --long_label LONG_LABEL`
  Long label for hub, e.g. english organism name, if it contains whitespaces, pass it with quotation marks: `---long_label "fruit fly"`

- `-l SHORT_LABEL, --short_label SHORT_LABEL`
  Short label (without whitespaces and special characters) for identifying assembly hub. The short label will also be used as assembly version, unless the following option is specified:

- `-V ASSEMBLY_VERSION, --assembly_version ASSEMBLY_VERSION`
  Assembly version, e.g. "BDGP R4/dm3". This argument must be provided if the hub is supposed to be added to the public UCSC list.

- `-N LATIN_NAME, --latin_name LATIN_NAME`
  Latin species name, e.g. "Drosophila melanogaster". This argument must be provided if the hub is supposed to be added to the public UCSC list.

- `-s SAMTOOLS_PATH, --SAMTOOLS_PATH SAMTOOLS_PATH`
  Path to samtools executable. By default, make_hub.py will search for a samtools executable in your $PATH. On some systems, e.g. high performance compute clusters, it may be more conventient to specify the path to samtools with this option while calling make_hub.py

- `-B BAM2WIG_PATH, --BAM2WIG_PATH BAM2WIG_PATH`
  Path to bam2wig executable. bam2wig from AUGUSTUS auxprogs is not required for converting a BAM to a WIG file with make_hub.py. It may be a little faster than the built-in conversion function, though. By default, make_hub.py will search for a bam2wig executable in your $PATH. On some systems, e.g. high performance compute clusters, it may be more conventient to specify the path to bam2wig with this option while calling make_hub.py

- `-b BAM [BAM ...], --bam BAM [BAM ...]`
  BAM file(s) - space separated - with RNA-Seq information, will be displayed as BigWig coverage track.

- `-d, --display_bam_as_bam`
  Display BAM file(s) as bam tracks (in addition to BigWig coverage tracks)

- `-c CORES, --cores CORES`
  Number of cores for samtools sort processes that are used for producing BAM tracks. Usage of more than one core may

significantly speed up track generation.

- `-a ANNOT, --annot ANNOT`
  GTF file with reference annotation (may be particularly interesting to visualize in case of re-annotation of genomes).

- `-X BRAKER_OUT_DIR, --braker_out_dir BRAKER_OUT_DIR`
  BRAKER output directory with GTF files. If this option is specified, the following options are set, automatically, using the files in BRAKER_OUT_DIR (if these files exist):

  - `-i HINTS, --hints HINTS`
  - `-t TRAINGENES, --traingenes TRAINGENES`
  - `-m GENEMARK, --genemark GENEMARK`
  - `-w AUG_AB_INITIO, --aug_ab_initio AUG_AB_INITIO`
  - `-x AUG_HINTS, --aug_hints AUG_HINTS`
  - `-y AUG_AB_INITIO_UTR, --aug_ab_initio_utr AUG_AB_INITIO_UTR`
  - `-z AUG_HINTS_UTR, --aug_hints_utr AUG_HINTS_UTR`

- `-i HINTS, --hints HINTS`
  GFF file with BRAKER hints (AUGUSTUS-specific GFF format of BRAKER).

- `-t TRAINGENES, --traingenes TRAINGENES`
  GTF file with training genes.

- `-m GENEMARK, --genemark GENEMARK`
  GTF file with GeneMark predictions.

- `-w AUG_AB_INITIO, --aug_ab_initio AUG_AB_INITIO`
  GTF file with ab initio AUGUSTUS predictions

- `-x AUG_HINTS, --aug_hints AUG_HINTS`
  GTF file with AUGUSTUS predictions with hints

- `-y AUG_AB_INITIO_UTR, --aug_ab_initio_utr AUG_AB_INITIO_UTR`
  GTF file with ab initio AUGUSTUS predictions with UTRs

- `-z AUG_HINTS_UTR, --aug_hints_utr AUG_HINTS_UTR`
  GTF file with AUGUSTUS predictions with hints with UTRs

- `-M MAKER_GFF, --maker_gff MAKER_GFF`
  MAKER2 output file in GFF3 format. This file could be the result of a `gff3_merge -d *_master_datastore_index.log` command.

- `-I GLIMMER_GFF, --glimmer_gff GLIMMER_GFF`
  GlimmerHMM output file in GFF3 format. This file could be the result of a `glimmhmm.pl glimmerhmm_linux_x86_64 genome.fa trained_dir/human "-g -o glimmer.out"` command.

- `-S SNAP_GFF, --snap_gff SNAP_GFF`
  SNAP output file in GFF3 format. This file could e.g. be the result of the two commands
  1) `snap worm genome.fa > snap.zff`
  2) `cat snap.zff | zff2gff3.pl > snap.gff`

- `-E GEMOMA_FILTERED_PREDICTIONS, --gemoma_filtered_predictions GEMOMA_FILTERED_PREDICTIONS`
  GFF3 output file of Gemoma (filtered_predictions.gff)

- `-G GENE_TRACK [GENE_TRACK ...], --gene_track GENE_TRACK [GENE_TRACK ...]`
  Gene track with user specified label, argument must be formatted as follows for adding a single track: `--gene_track file.gtf tracklabel`

- `-A, --add_track`
  Add track(s) to existing hub

- `-o OUTDIR, --outdir OUTDIR`
  Output directory to write hub to (default is the current working directory). This directory must be writable.

- `-r, --no_tmp_rm`
  Do not delete temporary files (e.g. for debugging purposes).

- `-P, --no_genePredToBigGenePred`
  Option for the special case in which the precompiled UCSC binaries are not working on your system, and you installed kentutils from the older ENCODE github repository; if activated, gene prediction tracks will be output to bigBed instead of bigGenePred format and amino acid display will not be possible.

- `-v VERBOSITY, --verbosity VERBOSITY`
  If INT VERBOSITY > 0, verbose logging output is produced (e.g. for debugging purposes).

# Example data

Example data is located in the directory `data/`. It consists of the following files:

- `genome.fa`: sequence LN902858_1 of *Hymenolepis microstoma*, assembly version GCA_000469805.2 from GenBank.

- `rnaseq.fa`: RNA-Seq reads of library ERR337976 that mapped to sequence LN902858_1 with Hisat2.

- `annot.gtf`: NCBI reference annotation of scaffold LN902858_1.

- `augustus.ab_initio.gtf`: AUGUSTUS *ab inito* gene predictions from a BRAKER run (run was performed on the complete genome, predictions corresponding to LN902858_1 were extracted) with Hisat2 alignments from RNA-Seq library ERR337976.

- `augustus.hints.gtf`: AUGUSTUS gene predictions with hints from a BRAKER run (run was performed on the complete genome, predictions corresponding to LN902858_1 were extracted) with Hisat2 alignments from RNA-Seq library ERR337976.

- `GeneMark-ET/genemark.gtf`: GeneMark-ES/ET predictions from a BRAKER run (run was performed on the complete genome, predictions corresponding to LN902858_1 were extracted) with Hisat2 alignments from RNA-Seq library ERR337976.

- `hintsfile.gff`: Hints from a BRAKER run (run was performed on the complete genome, hints corresponding to LN902858_1 were extracted) with Hisat2 alignments from RNA-Seq library ERR337976.

- `gemoma.gff`: Gemoma predictions from a Gemoma run with Hisat2 alignments from RNA-Seq library ERR337976 and proteins of *Echinococcus multilocularis*. (Run was performed on the complete genome, predictions corresponding to LN902858_1 were extracted)

- `maker.gff`: MAKER2 predictions from a run with BRAKER gene models as model_gff, Cufflinks assembly of Hisat2 alignments of RNA-Seq library ERR337976, a custom repeat library for RepeatMasker, AUGUSTUS with BRAKER-trained parameters, BUSCO predictions as evidence, and GeneMark-ES/ET predictions with BRAKER-trained parameters.

- `glimmer.gff`: GlimmerHMM predictions from running GlimmerHMM on target genome sequence with human parameters: `glimmhmm.pl glimmerhmm_linux_x86_64 genome.fa trained_dir/human "-g -o glimmer.out`

- `snap.gff`: SNAP prediction from running SNAP on target genome sequence with worm parameters, subsequent conversion from zff to gff-format with the SNAP script zff2gff3.pl:

```
snap worm genome.fa > snap.zff
cat snap.zff | zff2gff3.pl > snap.gff
```

# Output of MakeHub

make_hub.py produces a directory that is called identical to the argument for option `--short_label` / `-l`. Let's assume the short label had been `species`.

`species` contains the following files:

- `hub.txt` - this file contains basic information about the assembly hub, for example, the short and long labels, a reference to `genomes.txt`, and contact information. You might need to modify this file for adding it to UCSC's list of public hubs (see section Steps to take prior adding a MakeHub Track Data Hub to the UCSC public hub list).

- `genomes.txt` - this file contains references to the configuration files `trackDb.txt` and `groups.txt`, as well as for example a default browsing location.

- `aboutHub.html` - this file should contain a meaningful description of your assembly hub. Please edit this file, manually. You might need to modify this file for adding it to UCSC's list of public hubs (see section Steps to take prior adding a MakeHub Track Data Hub to the UCSC public hub list).

Furthermore, `species` contains another directory `species` in which the hub configuration files `trackDb.txt` and `groups.txt`, as well as all files that are required for browsing tracks, reside. The number of files may differ depending on how many tracks have actually been created.

Importantly, `species` also contains `*.html` files for all tracks. These files should be edited, manually, to contain meaningful information and you will have to edit them in order to add the hub to UCSC's list of public hubs (see section Steps to take prior adding a MakeHub Track Data Hub to the UCSC public hub list).

# How to use MakeHub output with UCSC Genome Browser

Copy the complete hub folder (e.g. `species`) to a publicly accessible web server.

Go to https://genome.ucsc.edu/index.html, click on `My Data` -> `Track Hubs` -> `My Hubs` and add the link to your publicly available `hub.txt` file into the URL window. Subsequently, click on `Add Hub`.

# Steps to take prior adding a MakeHub Track Data Hub to the UCSC public hub list

For running MakeHub, we use the value of argument `--short_label` as directory name for the genomic data of a species. Therefore, we prefer to have a 'short label' without spaces. UCSC, however, wants to a have real 'short description', e.g. consisting of multiple words as a short label. Therefore, go to the file `hub.txt` and change the shortLabel entry to something more descriptive. If you already had a very descriptive `--long_label`, you are good to go with that, but if you also didn't choose something very descriptive here, please edit.

For example, we chose the 'short label' "Purple sea star Assembly Hub" and the 'long label' "This is the Assembly Hub of the purple sea star (Pisaster ochraceus, assembly 12Jun2017_28pcJ)" for a public hub.

Further, edit all the description HTML pages of your hub and add information specific to the data that you are visualizing. The description pages are located in your hub directory. `aboutHub.html` is located at the top of the hub directory structure. You can find all other pages by calling

```
ls $shortLabel/$shortLabel/*.html
```

where $shortLabel is the actual shortLabel that you used to build the hub.

Please also have a look at http://genomewiki.ucsc.edu/index.php/Public_Hub_Guidelines prior asking the UCSC team to add your hub to the list of public hubs.

# Bug reporting

Before reporting bugs, please check that you are using the most recent versions of MakeHub. Also, check the open and closed issues on github at https://github.com/Gaius-Augustus/MakeHub/issues for possible solutions to your problem.

## Reporting bugs on github

If you found a bug, please open an issue at https://github.com/Gaius-Augustus/MakeHub/issues (or contact katharina.hoff@uni-greifswald.de).

Information worth mentioning in your bug report:

make_hub.py prints information about separate steps on STDOUT. Please let us know at which step and with what error message make_hub.py caused problems.

# Citing MakeHub

Hoff KJ, “MakeHub: Fully automated generation of UCSC Genome Browser Assembly Hubs.” *bioRxiv*: doi: https://doi.org/10.1101/550145

# License

All source code is under GNU public license 3.0 (see https://www.gnu.org/licenses/gpl-3.0.de.html).

# References

**[1]** Raney BJ, Dreszer TR, Barber GP, Clawson H, Fujita PA, Wang T, Nguyen N, Paten B, Zweig AS, Karolchik D, Kent WJ. 2014.
“Track Data Hubs.” *Bioinformatics* 1;30(7):1003-5.↩

**[2]** Kent WJ, Sugnet CW, Furey TS, Roskin KM, Pringle TH, Zahler AM, Haussler D. 2002.
“UCSC Genome Browser.” *Genome Res.* 12(6):996-1006.↩

**[3]** Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R. 2009.
“The sequence alignment/map format and SAMtools.” *Bioinformatics* 26(16):2078-2079.↩

**[4]** Stanke M, Diekhans M, Baertsch R, Haussler D. 2008.
“Using native and syntenically mapped cDNA alignments to improve de novo gene finding.” *Bioinformatics* 24(5):637-644.↩

**[5]** Hoff KJ, Lange S, Lomsadze A, Borodovsky M, Stanke M. 2015.
“BRAKER1: unsupervised RNA-Seq-based genome annotation with GeneMark-ET and AUGUSTUS.” *Bioinformatics* 32(5), 767-769.↩

**[6]** Holt C, Yandell M. 2011.
“MAKER2: an annotation pipeline and genome-database management tool for second-generation genome projects.” *BMC Bioinformatics* 12(1), 491.↩

**[7]** Keilwagen J, Hartung F, Paulini M, Twardziok SO, Grau J. 2018.
“Combining RNA-seq data and homology-based gene prediction for plants, animals and fungi.” *BMC Bioinformatics* 19(1), 189.↩

**[8]** Korf I. 2004.
“Gene finding in novel genomes” *BMC Bioinformatics* 5, 59.↩

**[9]** Majoros WH, Salzberg SL. 2004.
“TigrScan and GlimmerHMM: two open source ab initio eukaryotic gene-finders” *Bioinformatics* 20(16), 2878-2879.↩