
pyHML Documentation

Release 0.0.4

Mike Halagan

Aug 06, 2018

1	pyHML	3
1.1	Features	3
1.2	Install	4
1.3	Credits	4
2	Installation	5
2.1	Stable release	5
2.2	From sources	5
3	History	7
3.1	0.0.5 (2017-04-16)	7
3.2	0.0.4 (2017-04-15)	7
3.3	0.0.3 (2017-04-14)	7
3.4	0.0.2 (2017-11-14)	7
3.5	0.0.1 (2017-10-19)	8
4	pyhml package	9
4.1	pyhml	9
4.2	Data Objects	10
5	Contributing	19
5.1	Types of Contributions	19
5.2	Get Started!	20
5.3	Pull Request Guidelines	21
5.4	Tips	21
6	Usage	23
7	Indices and tables	25
	Python Module Index	27

Copyright (c) 2017 Be The Match operated by National Marrow Donor Program. All Rights Reserved.

Python HML parser

- Free software: LGPL 3.0
- Documentation: <https://pyhml.readthedocs.io>.
- Jupyter Notebook

1.1 Features

```
import pyhml
hml_file = "hml_example.xml"
hmlparser = pyhml.HmlParser()
hml = hmlparser.parse(hml_file)
outdir = 'output/directory'

# Print out each subject in fasta format
hml.tobiotype(outdir, dtype='fasta', by='subject')

# Print out the full HML file in IMGT dat file format
hml.tobiotype(outdir, dtype='imgt', by='file')

# Get pandas DF from HML object
pandasdf = hml.toPandas()
print(pandasdf)
```

#	ID	Locus	glstring	dbversion	\
# 0	1000-000-0	HLA-A	HLA-A*01:01:01+HLA-A*24:02:01	3.14.0	
# 1	1000-000-0	HLA-A	HLA-A*01:01:01+HLA-A*24:02:01	3.14.0	
# 2	1000-000-0	HLA-A	HLA-A*01:01:01+HLA-A*24:02:01	3.14.0	
# 3	1000-000-0	HLA-A	HLA-A*01:01:01+HLA-A*24:02:01	3.14.0	
# 4	1000-000-0	HLA-B	HLA-B*08:01:01+HLA-B*57:01:01	3.14.0	
# 5	1000-000-0	HLA-B	HLA-B*08:01:01+HLA-B*57:01:01	3.14.0	

(continues on next page)

(continued from previous page)

# 6	1000-000-0	HLA-B	HLA-B*08:01:01+HLA-B*57:01:01	3.14.0
# 7	1000-000-0	HLA-B	HLA-B*08:01:01+HLA-B*57:01:01	3.14.0
# 8	1000-000-0	HLA-C	HLA-C*06:02:01+HLA-C*07:01:01	3.14.0
# 9	1000-000-0	HLA-C	HLA-C*06:02:01+HLA-C*07:01:01	3.14.0
# 10	1000-000-0	HLA-C	HLA-C*06:02:01+HLA-C*07:01:01	3.14.0
# 11	1000-000-0	HLA-C	HLA-C*06:02:01+HLA-C*07:01:01	3.14.0
# 12	1000-000-0	HLA-DPB1	HLA-DPB1*02:01:02+HLA-DPB1*04:01:01	3.14.0
# 13	1000-000-0	HLA-DPB1	HLA-DPB1*02:01:02+HLA-DPB1*04:01:01	3.14.0
# 14	1000-000-0	HLA-DRB1	HLA-DRB1*03:01:01+HLA-DRB1*07:01:01	3.15.0
# 15	1000-000-0	HLA-DRB1	HLA-DRB1*03:01:01+HLA-DRB1*07:01:01	3.15.0
#	sequence			
# 0	TTCTGGATACTCACGACGCGGACCCAGTTCTCACTCCCATTGGGT...			
# 1	TTCCCGTCAGACCCCCCAAGACACATATGACCCACCACCCCATCT...			
# 2	TTCTGGATACTCACGACGCGGACCCAGTTCTCACTCCCATTGGGT...			
# 3	GTGCTGTGTCCAGGCTGGTGTCTGGGTCTGTGCTCTCTTCCCA...			
# 4	CCATGGTGAGTTTCCCTGTACAAGAGTCCAAGGGGAGAGGTAAGTG...			
# 5	GGCCTCTGCGGAGAGGAGCGAGGGGCCCGCCGGCGAGGGCGCAGG...			
# 6	CCATGGTGAGTTTCCCTGTACAAGAGTCCAAGGGGAGAGGTAAGTG...			
# 7	GGCCTCTGCGGAGAGGAGCGAGGGGCCCGCCGGCGAGGGCGCAGG...			
# 8	AGGGATCAGGACGAAGTCCCAGGTCCCGGACGGGGCTCTCAGGGTC...			
# 9	CGCATCCCCACTTCCCACTCCCATTGGGTGTGCGATATCTAGAGAA...			
# 10	AGGGATCAGGACGAAGTCCCAGGTCCCGGACGGGGCTCTCAGGGTC...			
# 11	CGCATCCCCACTTCCCACTCCCATTGGGTGTGCGATATCTAGAGAA...			
# 12	CCAATTGGCCAATTGGCCAATTGGCCAATTGGCCAATTGGCCAATT...			
# 13	CCAATTGGCCAATTGGCCAATTGGCCAATTGGCCAATTGGCCAATT...			
# 14	CATGCATGCATGCATGCATGCATGCATGCATGCATGCATGCATGCA...			
# 15	CATGCATGCATGCATGCATGCATGCATGCATGCATGCATGCATGCA...			

1.2 Install

```
pip install pyhml
```

1.3 Credits

This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.

2.1 Stable release

To install pyHML, run this command in your terminal:

```
$ pip install pyhml
```

This is the preferred method to install pyHML, as it will always install the most recent stable release.

If you don't have [pip](#) installed, this [Python installation guide](#) can guide you through the process.

2.2 From sources

The sources for pyHML can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/mhalagan-nmdp/pyhml
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/mhalagan-nmdp/pyhml/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```


3.1 0.0.5 (2017-04-16)

- Improved documentation
- Fixed issues with parsing HML files with NMDP-CORRECTION

3.2 0.0.4 (2017-04-15)

- Fixed dependency issues.
- Moved tobiotype to HML object.
- Moved toDF to HML object and renamed toPandas()
- Added tests and linked to travis.ci

3.3 0.0.3 (2017-04-14)

- Added the ability to parse .gz files
- Added the ability to parse HML files with bad tags.

3.4 0.0.2 (2017-11-14)

- Fixed issues with parsing HML files with missing data

3.5 0.0.1 (2017-10-19)

- First release on PyPI.

4.1 pyhml

class `pyhml.pyhml.HmlParser` (*hmlversion: str = None, verbose: bool = False*)

Bases: `object`

A python HML parser that converts any valid HML file into an python `object`. Allows users to easily interact with HML data as python objects. Users can also easily convert the HML data to a pandas `DataFrame`. If no `hmlversion` is provided, then the schemas for all HML versions are loaded.

Examples:

```
>>> import pyhml
>>> hmlparser = pyhml.HmlParser(verbose=True)
>>> hml = hmlparser.parse(hml_file)
>>> hml_df = hml.toPandas()
```

Parameters

- **hmlversion** (*str*) – A specific HML version to load.
- **verbose** (*bool*) – Flag for running in verbose.

parse (*hml_file: str*) → `pyhml.models.hml.HML`

Parses an HML file into a python object.

```
>>> hml = hmlparser.parse(hml_file)
```

Parameters `hml_file` – A valid HML file

Type `str`

Returns Object containing HML data

Return type *HML*

4.2 Data Objects

4.2.1 HML

```
class pyhml.models.hml.HML (project_name: str = None, version: str = None,
                             schema_location: str = None, reporting_center:
                             pyhml.models.reporting_center.ReportingCenter = None, sample:
                             List[pyhml.models.sample.Sample] = None)
```

Bases: `pyhml.models.base_model_.Model`

NOTE: This class is auto generated by the swagger code generator program. Do not edit the class manually.

```
classmethod from_dict (dikt) → pyhml.models.hml.HML
```

Returns the dict as a model

Parameters `dikt` – A dict.

Type dict

Returns The HML of this HML.

Return type *HML*

project_name

Gets the project_name of this HML.

Returns The project_name of this HML.

Return type str

reporting_center

Gets the reporting_center of this HML.

Returns The reporting_center of this HML.

Return type ReportingCenter

sample

Gets the sample of this HML.

Returns The sample of this HML.

Return type List[*Sample*]

schema_location

Gets the schema_location of this HML.

Returns The schema_location of this HML.

Return type str

toPandas () → pandas.core.frame.DataFrame

Returns all the HML data as a pandas DataFrame.

Examples:

```
>>> import pyhml
>>> hmlparser = pyhml.HmlParser(verbose=True)
>>> hml = hmlparser.parse(hml_file)
>>> hml_df = hml.toPandas()
```

Returns Pandas dataframe

Return type DataFrame

tobiotype (*outdir*, *dtype*='fasta', *by*='file')

Converts an HML object to a BioPython data format

Examples:

```
>>> import pyhml
>>> hmlparser = pyhml.HmlParser(verbose=True)
>>> hml = hmlparser.parse(hml_file)
>>> hml.tobiotype("output/directory", dtype='imgt', by='subject')
```

Parameters

- **outdir** (*str*) – The output directory
- **dtype** – The BioPython output type
- **by** (*str*) – What to print out the HML file by

version

Gets the version of this HML.

Returns The version of this HML.

Return type str

4.2.2 Sample

class pyhml.models.sample.**Sample** (*center_code*: int = None, *id*: str = None, *collection_method*: str = None, *typing*: List[pyhml.models.typing.Typing] = None)

Bases: pyhml.models.base_model_.Model

Examples:

```
>>> from pyhml.models.typing import Typing
>>> from pyhml.models.sample import Sample
```

center_code

Gets the center_code of this Sample.

Returns The center_code of this Sample.

Return type int

collection_method

Gets the collection_method of this Sample.

Returns The collection_method of this Sample.

Return type str

create_seqrecords ()

Creates the seq_records for this Sample.

classmethod **from_dict** (*dikt*) → pyhml.models.sample.Sample

Returns the dict as a model

Parameters **dikt** – A dict.

Type dict

Returns The Sample of this Sample.

Return type *Sample*

id

Gets the id of this Sample.

Returns The id of this Sample.

Return type str

seq_records

Gets the seq_records of this Sample.

Returns The seq_records of this Sample.

Return type Dict

typing

Gets the typing of this Sample.

Returns The typing of this Sample.

Return type List[*Typing*]

4.2.3 Typing

```
class pyhml.models.typing.Typing (date:          str          = None,      gene_family:
                                     str          = None,      allele_assignment:
                                     List[pyhml.models.allele_assignment.AlleleAssignment]
                                     = None,      consensus_sequence:
                                     List[pyhml.models.consensus.Consensus] = None, typ-
                                     ing_method: pyhml.models.typing_method.TypingMethod =
                                     None)
Bases: pyhml.models.base_model_.Model
```

NOTE: This class is auto generated by the swagger code generator program. Do not edit the class manually.

allele_assignment

Gets the allele_assignment of this Typing.

Returns The allele_assignment of this Typing.

Return type List[*AlleleAssignment*]

consensus_sequence

Gets the consensus_sequence of this Typing.

Returns The consensus_sequence of this Typing.

Return type List[*Consensus*]

create_seqrecord (*subid*)

date

Gets the date of this Typing.

Returns The date of this Typing.

Return type str

classmethod from_dict (*dikt*) → pyhml.models.typing.Typing

Returns the dict as a model

Parameters *dikt* – A dict.

Type dict

Returns The Typing of this Typing.

Return type *Typing*

gene_family

Gets the gene_family of this Typing.

Returns The gene_family of this Typing.

Return type str

seq_records

Gets the seq_records of this ReferenceData.

Returns The seq_records of this ReferenceData.

Return type List[SeqRecord]

typing_method

Gets the typing_method of this Typing.

Returns The typing_method of this Typing.

Return type TypingMethod

4.2.4 Consensus

```
class pyhml.models.consensus.Consensus (date: str = None, consensus_sequence_block:
                                         List[pyhml.models.consensus_seq_block.ConsensusSeqBlock]
                                         = None, reference_database:
                                         List[pyhml.models.ref_database.RefDatabase]
                                         = None)
```

Bases: pyhml.models.base_model_.Model

NOTE: This class is auto generated by the swagger code generator program. Do not edit the class manually.

consensus_sequence_block

Gets the consensus_sequence_block of this Consensus.

Returns The consensus_sequence_block of this Consensus.

Return type List[*ConsensusSeqBlock*]

date

Gets the date of this Consensus.

Returns The date of this Consensus.

Return type str

classmethod from_dict (dikt) → pyhml.models.consensus.Consensus

Returns the dict as a model

Parameters **dikt** – A dict.

Type dict

Returns The Consensus of this Consensus.

Return type *Consensus*

reference_database

Gets the reference_database of this Consensus.

Returns The reference_database of this Consensus.

Return type List[RefDatabase]

4.2.5 Consensus Block

```
class pyhml.models.consensus_seq_block.ConsensusSeqBlock (continuity: bool = None,  
description: str = None,  
end: int = None, expected_copy_number:  
int = None, phase_set:  
str = None, reference_sequence_id: str =  
None, start: int = None,  
strand: str = None,  
sequence: Bio.Seq.Seq =  
None, sequence_quality:  
List[pyhml.models.seq_quality.SeqQuality]  
= None, variant:  
List[pyhml.models.variant.Variant]  
= None)
```

Bases: pyhml.models.base_model_.Model

NOTE: This class is auto generated by the swagger code generator program. Do not edit the class manually.

continuity

Gets the continuity of this ConsensusSeqBlock.

Returns The continuity of this ConsensusSeqBlock.

Return type bool

description

Gets the description of this ConsensusSeqBlock.

Returns The description of this ConsensusSeqBlock.

Return type str

end

Gets the end of this ConsensusSeqBlock.

Returns The end of this ConsensusSeqBlock.

Return type int

expected_copy_number

Gets the expected_copy_number of this ConsensusSeqBlock.

Returns The expected_copy_number of this ConsensusSeqBlock.

Return type int

classmethod from_dict (dikt) → pyhml.models.consensus_seq_block.ConsensusSeqBlock

Returns the dict as a model

Parameters dikt – A dict.

Type dict

Returns The ConsensusSeqBlock of this ConsensusSeqBlock.

Return type *ConsensusSeqBlock*

phase_set

Gets the phase_set of this ConsensusSeqBlock.

Returns The phase_set of this ConsensusSeqBlock.

Return type str

reference_sequence_id

Gets the reference_sequence_id of this ConsensusSeqBlock.

Returns The reference_sequence_id of this ConsensusSeqBlock.

Return type str

sequence

Gets the sequence of this ConsensusSeqBlock.

Returns The sequence of this ConsensusSeqBlock.

Return type Seq

sequence_quality

Gets the sequence_quality of this ConsensusSeqBlock.

Returns The sequence_quality of this ConsensusSeqBlock.

Return type List[SeqQuality]

start

Gets the start of this ConsensusSeqBlock.

Returns The start of this ConsensusSeqBlock.

Return type int

strand

Gets the strand of this ConsensusSeqBlock.

Returns The strand of this ConsensusSeqBlock.

Return type str

variant

Gets the variant of this ConsensusSeqBlock.

Returns The variant of this ConsensusSeqBlock.

Return type List[Variant]

4.2.6 Allele Assignment

```
class pyhml.models.allele_assignment.AlleleAssignment(allele_db: str = None, allele_version: str = None, date: str = None, glstring: List[str] = None, haploid: List[pyhml.models.haploid.Haploid] = None)
```

Bases: pyhml.models.base_model_.Model

NOTE: This class is auto generated by the swagger code generator program. Do not edit the class manually.

allele_db

Gets the allele_db of this AlleleAssignment.

Returns The allele_db of this AlleleAssignment.

Return type str

allele_version

Gets the allele_version of this AlleleAssignment.

Returns The allele_version of this AlleleAssignment.

Return type str

date

Gets the date of this AlleleAssignment.

Returns The date of this AlleleAssignment.

Return type str

classmethod from_dict (*dikt*) → pyhml.models.allele_assignment.AlleleAssignment

Returns the dict as a model

Parameters **dikt** – A dict.

Type dict

Returns The AlleleAssignment of this AlleleAssignment.

Return type *AlleleAssignment*

glstring

Gets the glstring of this AlleleAssignment.

Returns The glstring of this AlleleAssignment.

Return type List[str]

haploid

Gets the haploid of this AlleleAssignment.

Returns The haploid of this AlleleAssignment.

Return type List[Haploid]

4.2.7 Allele Assignment

```
class pyhml.models.allele_assignment.AlleleAssignment (allele_db: str = None, allele_version: str = None, date: str = None, glstring: List[str] = None, haploid: List[pyhml.models.haploid.Haploid] = None)
```

Bases: pyhml.models.base_model_.Model

NOTE: This class is auto generated by the swagger code generator program. Do not edit the class manually.

allele_db

Gets the allele_db of this AlleleAssignment.

Returns The allele_db of this AlleleAssignment.

Return type str

allele_version

Gets the allele_version of this AlleleAssignment.

Returns The allele_version of this AlleleAssignment.

Return type str

date

Gets the date of this AlleleAssignment.

Returns The date of this AlleleAssignment.

Return type str

classmethod from_dict (*dikt*) → pyhml.models.allele_assignment.AlleleAssignment

Returns the dict as a model

Parameters **dikt** – A dict.

Type dict

Returns The AlleleAssignment of this AlleleAssignment.

Return type *AlleleAssignment*

glstring

Gets the glstring of this AlleleAssignment.

Returns The glstring of this AlleleAssignment.

Return type List[str]

haploid

Gets the haploid of this AlleleAssignment.

Returns The haploid of this AlleleAssignment.

Return type List[Haploid]

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at <https://github.com/mhalagan-nmdp/pyhtml/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

5.1.4 Write Documentation

pyHML could always use more documentation, whether as part of the official pyHML docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/mhalagan-nmdp/pyhml/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up *pyhml* for local development.

1. Fork the *pyhml* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/pyhml.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv pyhml
$ cd pyhml/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 pyhml tests
$ python setup.py test or py.test
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/mhalagan-nmdp/pyhml/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_pyhml
```

Usage

To use pyHML in a project:

```
import pyhml
    hmlparser = pyhml.HmlParser()
hml = hmlparser.parse("hml_example.xml")
pandasdf = hml.toPandas()

# Ouput the HML data as a IPD-IMGT/HLA .dat file for each subject
hml.tobiotype("output/directory", dtype='imgt', by='subject')

# Output the whole HML file as one fasta file
hml.tobiotype("output/directory", dtype='fasta', by='file')

# Defaults to dtype='fasta' and by='subject'
hml.tobiotype("output/directory")
```


CHAPTER 7

Indices and tables

- `genindex`
- `modindex`
- `search`

p

- `pyhml.models.allele_assignment`, [16](#)
- `pyhml.models.consensus`, [13](#)
- `pyhml.models.consensus_seq_block`, [14](#)
- `pyhml.models.hml`, [10](#)
- `pyhml.models.sample`, [11](#)
- `pyhml.models.typing`, [12](#)
- `pyhml.pyhml`, [9](#)

A

allele_assignment (pyhml.models.typing.Typing attribute), 12
 allele_db (pyhml.models.allele_assignment.AlleleAssignment attribute), 15, 16
 allele_version (pyhml.models.allele_assignment.AlleleAssignment attribute), 16
 AlleleAssignment (class in pyhml.models.allele_assignment), 15, 16

C

center_code (pyhml.models.sample.Sample attribute), 11
 collection_method (pyhml.models.sample.Sample attribute), 11
 Consensus (class in pyhml.models.consensus), 13
 consensus_sequence (pyhml.models.typing.Typing attribute), 12
 consensus_sequence_block (pyhml.models.consensus.Consensus attribute), 13
 ConsensusSeqBlock (class in pyhml.models.consensus_seq_block), 14
 continuity (pyhml.models.consensus_seq_block.ConsensusSeqBlock attribute), 14
 create_seqrecord() (pyhml.models.typing.Typing method), 12
 create_seqrecords() (pyhml.models.sample.Sample method), 11

D

date (pyhml.models.allele_assignment.AlleleAssignment attribute), 16, 17
 date (pyhml.models.consensus.Consensus attribute), 13
 date (pyhml.models.typing.Typing attribute), 12
 description (pyhml.models.consensus_seq_block.ConsensusSeqBlock attribute), 14

E

end (pyhml.models.consensus_seq_block.ConsensusSeqBlock attribute), 14

expected_copy_number (pyhml.models.consensus_seq_block.ConsensusSeqBlock attribute), 14

F

from_dict() (pyhml.models.allele_assignment.AlleleAssignment class method), 16, 17
 from_dict() (pyhml.models.consensus.Consensus class method), 13
 from_dict() (pyhml.models.consensus_seq_block.ConsensusSeqBlock class method), 14
 from_dict() (pyhml.models.hml.HML class method), 10
 from_dict() (pyhml.models.sample.Sample class method), 11
 from_dict() (pyhml.models.typing.Typing class method), 12

G

gene_family (pyhml.models.typing.Typing attribute), 13
 glstring (pyhml.models.allele_assignment.AlleleAssignment attribute), 16, 17

H

HML (class in pyhml.models.hml), 10
 HmlParser (class in pyhml.pyhml), 9

I

id (pyhml.models.sample.Sample attribute), 12

P

parse() (pyhml.pyhml.HmlParser method), 9
 phase_set (pyhml.models.consensus_seq_block.ConsensusSeqBlock attribute), 14
 project_name (pyhml.models.hml.HML attribute), 10
 pyhml.models.allele_assignment (module), 15, 16
 pyhml.models.consensus (module), 13
 pyhml.models.consensus_seq_block (module), 14
 pyhml.models.hml (module), 10

pyhml.models.sample (module), [11](#)
pyhml.models.typing (module), [12](#)
pyhml.pyhml (module), [9](#)

R

reference_database (pyhml.models.consensus.Consensus
attribute), [13](#)
reference_sequence_id (pyhml.models.consensus_seq_block.ConsensusSeqBlock
attribute), [15](#)
reporting_center (pyhml.models.hml.HML attribute), [10](#)

S

Sample (class in pyhml.models.sample), [11](#)
sample (pyhml.models.hml.HML attribute), [10](#)
schema_location (pyhml.models.hml.HML attribute), [10](#)
seq_records (pyhml.models.sample.Sample attribute), [12](#)
seq_records (pyhml.models.typing.Typing attribute), [13](#)
sequence (pyhml.models.consensus_seq_block.ConsensusSeqBlock
attribute), [15](#)
sequence_quality (pyhml.models.consensus_seq_block.ConsensusSeqBlock
attribute), [15](#)
start (pyhml.models.consensus_seq_block.ConsensusSeqBlock
attribute), [15](#)
strand (pyhml.models.consensus_seq_block.ConsensusSeqBlock
attribute), [15](#)

T

tobiotype() (pyhml.models.hml.HML method), [11](#)
toPandas() (pyhml.models.hml.HML method), [10](#)
Typing (class in pyhml.models.typing), [12](#)
typing (pyhml.models.sample.Sample attribute), [12](#)
typing_method (pyhml.models.typing.Typing attribute),
[13](#)

V

variant (pyhml.models.consensus_seq_block.ConsensusSeqBlock
attribute), [15](#)
version (pyhml.models.hml.HML attribute), [11](#)