python-hl7 Documentation

Release 0.2.2

John Paulett

August 18, 2014

Contents

python-hl7 is a simple library for parsing messages of Health Level 7 (HL7) version 2.x into Python objects. python-hl7 includes a simple client that can send HL7 messages to a Minimal Lower Level Protocol (MLLP) server (*mllp_send*).

HL7 is a communication protocol and message format for health care data. It is the de-facto standard for transmitting data between clinical information systems and between clinical devices. The version 2.x series, which is often is a pipe delimited format is currently the most widely accepted version of HL7 (version 3.0 is an XML-based format).

python-hl7 currently only parses HL7 version 2.x messages into an easy to access data structure. The current implementation does not completely follow the HL7 specification, but is good enough to parse the most commonly seen HL7 messages. The library could potentially evolve into being fully complainant with the spec. The library could eventually also contain the ability to create HL7 v2.x messages.

python-hl7 parses HL7 into a series of wrapped `hl7.Container` objects. The there are specific subclasses of `hl7.Container` depending on the part of the HL7 message. The `hl7.Container` message itself is a subclass of a Python list, thus we can easily access the HL7 message as an n-dimensional list. Specifically, the subclasses of `hl7.Container`, in order, are `hl7.Message`, `hl7.Segment`, and `hl7.Field`. Eventually additional containers will be added to fully support the HL7 specification.

CHAPTER 1

Usage

As an example, let's create a HL7 message:

```
>>> message = 'MSH|^~\&|GHH LAB|ELAB-3|GHH OE|BLDG4|200202150930||ORU^R01|CNTRL-3456|P|2.4\r'
>>> message += 'PID|||555-44-4444||EVERYWOMAN^EVE^E^^^^L|JONES|196203520|F|||153 FERNWOOD DR.^^STATES
>>> message += 'OBR|1|845439^GHH OE|1045813^GHH LAB|1554-5^GLUCOSE|||200202150730|||||||555-55-5555
>>> message += 'OBX|1|SN|1554-5^GLUCOSE^POST 12H CFST:MCNC:PT:SER/PLAS:QN||^182|mg/dl|70_105|H|||F'
```

We call the `hl7.parse()` command with string message:

```
>>> import hl7
>>> h = hl7.parse(message)
```

We get a `hl7.Message` object, wrapping a series of `hl7.Segment` objects:

```
>>> type(h)
<class 'hl7.Message'>
```

We can always get the HL7 message back:

```
>>> unicode(h) == message
True
```

Interestingly, `hl7.Message` can be accessed as a list:

```
>>> isinstance(h, list)
True
```

There were 4 segments (MSH, PID, OBR, OBX):

```
>>> len(h)
4
```

We can extract the `hl7.Segment` from the `hl7.Message` instance:

```
>>> h[3]
[[u'OBX'], [u'1'], [u'SN'], [u'1554-5', u'GLUCOSE', u'POST 12H CFST:MCNC:PT:SER/PLAS:QN'], [u''], [u'
```

We can easily reconstitute this segment as HL7, using the appropriate separators:

```
>>> unicode(h[3])
u'OBX|1|SN|1554-5^GLUCOSE^POST 12H CFST:MCNC:PT:SER/PLAS:QN||^182|mg/dl|70_105|H|||F'
```

We can extract individual elements of the message:

```
>>> h[3][3][1]
u'GLUCOSE'
>>> h[3][5][1]
u'182'
```

We can look up segments by the segment identifier, either via `hl7.Message.segments()` or via the traditional dictionary syntax:

```
>>> h.segments('OBX')[0][3][1]
u'GLUCOSE'
>>> h['OBX'][0][3][1]
u'GLUCOSE'
```

Since many many types of segments only have a single instance in a message (e.g. PID or MSH), `hl7.Message.segment()` provides a convienance wrapper around `hl7.Message.segments()` that returns the first matching `hl7.Segment`:

```
>>> h.segment('PID')[3][0]
u'555-44-4444'
```

CHAPTER 2

MLLP network client - `mllp_send`

python-hl7 features a simple network client, `mllp_send`, which reads HL7 messages from a file or `sys.stdin` and posts them to an MLLP server. `mllp_send` is a command-line wrapper around `hl7.client.MLLPClient`. `mllp_send` is a useful tool for testing HL7 interfaces or resending logged messages:

```
mllp_send --file sample.hl7 --port 6661 mirth.example.com
```

See *mllp_send - MLLP network client* for examples and usage instructions.

CHAPTER 3

Contents

3.1 python-hl7 API

hl7.**parse**(*line*)

Returns a instance of the hl7.Message that allows indexed access to the data elements.

Note: HL7 usually contains only ASCII, but can use other character sets (HL7 Standards Document, Section 1.7.1). Therefore, python-hl7 works on Python unicode strings. hl7.parse() will accept ASCII-only strings and automatically convert them into unicode. However, if the message contains non-ASCII characters, it is the responsibility of the caller of hl7.parse() to properly convert the message string to unicode first.

```
>>> h = hl7.parse(message)
```

Return type hl7.Message

hl7.**ishl7**(*line*)

Determines whether a *line* looks like an HL7 message. This method only does a cursory check and does not fully validate the message.

Return type bool

3.1.1 Data Types

class hl7.**Container**(*separator*, *sequence*=[ ])

Abstract root class for the parts of the HL7 message.

__unicode__()

Join a the child containers into a single string, separated by the self.separator. This method acts recursively, calling the children's __unicode__ method. Thus unicode() is the approriate method for turning the python-hl7 representation of HL7 into a standard string.

```
>>> unicode(h) == message
True
```

class hl7.**Message**(*separator*, *sequence*=[ ])

Representation of an HL7 message. It contains a list of hl7.Segment instances.

__getitem__(*key*)

Index or segment-based lookup.

If key is an integer, __getitem__ acts list a list, returning the hl7.Segment held at that index:

```
>>> h[1]
[[u'PID'], ...]
```

If the key is a string, `__getitem__` acts like a dictionary, returning all segments whose *segment_id* is *key* (alias of `hl7.Message.segments()`).

```
>>> h['OBX']
[[[u'OBX'], [u'1'], ...]]
```

Return type `hl7.Segment` or list of `hl7.Segment`

segment(*segment_id*)
Gets the first segment with the *segment_id* from the parsed *message*.

```
>>> h.segment('PID')
[[u'PID'], ...]
```

Return type `hl7.Segment`

segments(*segment_id*)
Returns the requested segments from the parsed *message* that are identified by the *segment_id* (e.g. OBR, MSH, ORC, OBX).

```
>>> h.segments('OBX')
[[[u'OBX'], [u'1'], ...]]
```

Return type list of `hl7.Segment`

class `hl7.`**Segment**(*separator*, *sequence*=[])
Second level of an HL7 message, which represents an HL7 Segment. Traditionally this is a line of a message that ends with a carriage return and is separated by pipes. It contains a list of `hl7.Field` instances.

class `hl7.`**Field**(*separator*, *sequence*=[])
Third level of an HL7 message, that traditionally is surrounded by pipes and separated by carets. It contains a list of strings.

3.1.2 MLLP Network Client

class `hl7.client.`**MLLPClient**(*host*, *port*)
A basic, blocking, HL7 MLLP client based upon `socket`.

MLLPClient implements two methods for sending data to the server.

- `MLLPClient.send()` for raw data that already is wrapped in the appropriate MLLP container (e.g. *<SB>message<EB><CR>*).
- `MLLPClient.send_message()` will wrap the message in the MLLP container

Can be used by the `with` statement to ensure `MLLPClient.close()` is called:

```
with MLLPClient(host, port) as client:
    client.send_message('MSH|...')
```

close()
Release the socket connection

send(*data*)
Low-level, direct access to the socket.send (data must be already wrapped in an MLLP container). Blocks until the server returns.

send_message(*message*)
Wraps a str, unicode, or **:py:cls:'hl7.Message'** in a MLLP container and send the message to the server

3.2 `mllp_send` - MLLP network client

python-hl7 features a simple network client, `mllp_send`, which reads HL7 messages from a file or `sys.stdin` and posts them to an MLLP server. `mllp_send` is a command-line wrapper around `hl7.client.MLLPClient`. `mllp_send` is a useful tool for testing HL7 interfaces or resending logged messages:

```
$ mllp_send --file sample.hl7 --port 6661 mirth.example.com
MSH|^~\&|LIS|Example|Hospital|Mirth|20111207105244||ACK^A01|A234244|P|2.3.1|
MSA|AA|234242|Message Received Successfully|
```

3.2.1 Usage

```
Usage: mllp_send [options] <server>

Options:
  -h, --help            show this help message and exit
  -p PORT, --port=PORT  port to connect to
  -f FILE, --file=FILE  read from FILE instead of stdin
  -q, --quiet           do not print status messages to stdout
  --loose               allow file to be a HL7-like object (\r\n instead of
                        \r). Can ONLY send 1 message. Requires --file option
                        (no stdin)
```

3.2.2 Input Format

By default, `mllp_send` expects the `FILE` or stdin input to be a properly formatted HL7 message (carriage returns separating segments) wrapped in a MLLP stream (`<SB>message1<EB><CR><SB>message2<EB><CR>...`).

However, it is common, especially if the file has been manually edited in certain text editors, that the ASCII control characters will be lost and the carriage returns will be replaced with the platform's default line endings. In this case, `mllp_send` provides the `--loose` option, which attempts to take something that "looks like HL7" and convert it into a proper HL7 message. Currently the `--loose` option can only handle 1 HL7 message per file (it causes `mllp_send` to assume the whole file is one HL7 message).

3.2.3 Additional Resources

- http://python-hl7.readthedocs.org

3.3 Contributing

The source code is available at http://github.com/johnpaulett/python-hl7

Please fork and issue pull requests. Generally any changes, bug fixes, or new features should be accompanied by corresponding tests in our test suite.

3.3.1 Testing

The test suite is located in `tests/` and can be run via `setup.py`:

```
$ python setup.py test
...
----------------------------------------------------------------------
Ran 17 tests in 0.005s

OK
```

Make sure the documentation is still valid:

```
$ pushd docs && make html man doctest && popd
...
Doctest summary
===============
   23 tests
    0 failures in tests
    0 failures in setup code
...
```

3.4 Change Log

3.4.1 0.2.2 - 2011-12-17

- *mllp_send* now takes the `--loose` options, which allows sending HL7 messages that may not exactly meet the standard (Windows newlines separating segments instead of carriage returns).

3.4.2 0.2.1 - 2011-08-30

- Added MLLP client (`hl7.client.MLLPClient`) and command line tool, *mllp_send*.

3.4.3 0.2.0 - 2011-06-12

- Converted `hl7.segment` and `hl7.segments` into methods on `hl7.Message`.
- Support dict-syntax for getting Segments from a Message (e.g. `message['OBX']`)
- Use unicode throughout python-hl7 since the HL7 spec allows non-ASCII characters. It is up to the caller of `hl7.parse()` to convert non-ASCII messages into unicode.
- Refactored from single hl7.py file into the hl7 module.
- Added Sphinx documentation. Moved project to github.

3.4.4 0.1.1 - 2009-06-27

- Apply Python 3 trove classifier

3.4.5 0.1.0 - 2009-03-13

- Support message-defined separation characters
- Message, Segment, Field classes

3.4.6 0.0.3 - 2009-01-09

- Initial release

3.5 Authors

John Paulett <john -at- paulett.org>

3.6 License

Copyright (C) 2009-2011 John Paulett (john -at- paulett.org)
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

 1. Redistributions of source code must retain the above copyright
    notice, this list of conditions and the following disclaimer.
 2. Redistributions in binary form must reproduce the above copyright
    notice, this list of conditions and the following disclaimer in
    the documentation and/or other materials provided with the
    distribution.
 3. The name of the author may not be used to endorse or promote
    products derived from this software without specific prior
    written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS
OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY
DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE
GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER
IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN
IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

CHAPTER 4

Install

python-hl7 is available on PyPi via `pip` or `easy_install`:

```
pip install -U hl7
```

For recent versions of Debian and Ubuntu, the *python-hl7* package is available:

```
sudo apt-get install python-hl7
```

CHAPTER 5

Links

- Documentation: http://python-hl7.readthedocs.org
- Source Code: http://github.com/johnpaulett/python-hl7
- PyPi: http://pypi.python.org/pypi/hl7

HL7 References:

- Health Level 7 - Wikipedia
- nule.org's Introduction to HL7
- hl7.org
- OpenMRS's HL7 documentation
- Transport Specification: MLLP