s2protocol Documentation

Release 1.0.1-dev

Blizzard Entertainment

Nov 14, 2017

Contents

1 Supported Versions **3**
1.1 Getting Started with s2protocol 3

2 Reference **5**
2.1 s2protocol Reference 5

3 Indices and tables **7**

Python Module Index **9**

`s2protocol` is a reference Python library and standalone tool to decode StarCraft II replay files into Python data structures.

Currently, `s2protocol` can decode these structures and events:

- replay header
- game details
- replay init data
- game events
- message events
- tracker events

`s2protocol` can be used as a base-build-specific library to decode binary blobs, or it can be run as a standalone tool to pretty print information from supported replay files.

Note that `s2protocol` does not expose game balance information or provide any kind of high level analysis of replays; it's meant to be just the first tool in the chain for your data mining application.

CHAPTER 1

Supported Versions

s2protocol supports all StarCraft II replay files that were written with retail versions of the game. The current plan is to support all future publicly released versions, including public betas.

1.1 Getting Started with s2protocol

1.1.1 Installing s2protocol

The easiest way to get started with s2protocol is to install it off PyPI using one of the following commands. Using your terminal, simply run either:

```
pip install s2protocol
```

or...:

```
easy_install s2protocol
```

Alternative Installs

If you'd prefer to download the package yourself, download it then run the following:

```
cd s2protocol
python setup.py install
```

Finally, if you'd like to help with the development of s2protocol, you can clone it directly from GitHub & setup a development install using:

```
git clone https://github.com/Blizzard/s2protocol.git
cd s2protocol
python setup.py develop
```

1.1.2 Using s2protocol Command Line Tool

If you're just interested in seeing the stats from a given replay file, `s2protocol` can be used in a standalone manner. A command line utility, called `s2_cli.py` is installed when the package is installed. Usage looks like:

```
s2_cli.py ~/Desktop/my_pvp_replay.SC2Replay --stats
```

This will output the stats from the replay file then exit.

Other options you can pass in include:

- `--gameevents`: Prints out the game events
- `--messageevents`: Prints out the message events
- `--trackerevents`: Prints out the tracker events
- `--attributeevents`: Prints out the attributes events
- `--header`: Prints out the protocol header
- `--details`: Prints out the protocol details
- `--initdata`: Prints out the protocol initdata
- `--stats`: Prints out the stats

To work inspect different protocol versions:

- `--versions`: Show all protocol versions
- `--diff`: Diff two protocol versions

To control the output format:

- `--json`: Outputs events as JSON structured documents
- `--ndjson`: Like –json but is each event is newline delimited

1.1.3 Using s2protocol Programmatically

If you're looking to do more than just look at a single replay's output once, you'll want to explore the programmatic API. Loading a replay & printing its header looks like:

```
>>> import mpyq

# Using mpyq, load the replay file.
>>> archive = mpyq.MPQArchive('~/Desktop/my_pvp_replay.SC2Replay')
>>> contents = archive.header['user_data_header']['content']

# Now parse the header information.
>>> from s2protocol import versions
>>> header = versions.latest().decode_replay_header(contents)
```

CHAPTER 2

Reference

Contents:

2.1 s2protocol Reference

2.1.1 s2_cli.py

2.1.2 decoders

class s2protocol.decoders.**BitPackedBuffer**(*contents*, *endian='big'*)

byte_align()

done()

read_aligned_bytes(*bytes*)

read_bits(*bits*)

read_unaligned_bytes(*bytes*)

used_bits()

class s2protocol.decoders.**BitPackedDecoder**(*contents*, *typeinfos*)

byte_align()

done()

instance(*typeid*)

used_bits()

exception s2protocol.decoders.**CorruptedError**

exception s2protocol.decoders.**TruncatedError**

class s2protocol.decoders.**VersionedDecoder**(*contents*, *typeinfos*)

byte_align()

done()

instance(*typeid*)

used_bits()

2.1.3 versions

2.1.4 protocol49716

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

s2protocol.decoders, 5

Index

B

BitPackedBuffer (class in s2protocol.decoders), 5
BitPackedDecoder (class in s2protocol.decoders), 5
byte_align() (s2protocol.decoders.BitPackedBuffer method), 5
byte_align() (s2protocol.decoders.BitPackedDecoder method), 5
byte_align() (s2protocol.decoders.VersionedDecoder method), 6

C

CorruptedError, 5

D

done() (s2protocol.decoders.BitPackedBuffer method), 5
done() (s2protocol.decoders.BitPackedDecoder method), 5
done() (s2protocol.decoders.VersionedDecoder method), 6

I

instance() (s2protocol.decoders.BitPackedDecoder method), 5
instance() (s2protocol.decoders.VersionedDecoder method), 6

R

read_aligned_bytes() (s2protocol.decoders.BitPackedBuffer method), 5
read_bits() (s2protocol.decoders.BitPackedBuffer method), 5
read_unaligned_bytes() (s2protocol.decoders.BitPackedBuffer method), 5

S

s2protocol.decoders (module), 5

T

TruncatedError, 5

U

used_bits() (s2protocol.decoders.BitPackedBuffer method), 5
used_bits() (s2protocol.decoders.BitPackedDecoder method), 5
used_bits() (s2protocol.decoders.VersionedDecoder method), 6

V

VersionedDecoder (class in s2protocol.decoders), 6