
xrootdlib Documentation

Release 0.2.0

Max Fischer

Sep 23, 2021

Library Overview

1	xrootdlib	1
2	Contributing and Feedback	23
	Python Module Index	25
	Index	27

1.1 xrootdlib package

1.1.1 Subpackages

xrootdlib.streams package

Subpackages

xrootdlib.streams.XrdXrootdMon package

Stream of the XrdXrootdMon monitoring messages

This module presents a high-level stream representation of the *XrdXrootdMon* struct primitives. It provides two stream mechanisms of different complexity:

- *stream_packets* provides an ordered stream of raw *Packet*
- *map_streams* provides a stream of *fstat*, *redir*, *trace*, and *map*

class stream_packets (*slave_args, **slave_kwargs)

Bases: chainlet.genlink.GeneratorLink

Provide a stream of packets from a readable bytes buffer

Parameters

- **packet_source** – a bytes buffer providing serialised packet data
- **sort_window** – window size in which to sort packets by their assigned order

The *packet_source* may be any bytes buffer supporting *read* operations. This can be an open file, but also a socket or wrapped memory. While the buffer may be extended indefinitely, it should *always* contain complete packets.

The `sort_window` is required to ensure ordering of the packet data. Since XRootD uses several packet buffers, concurrent packets may arrive in arbitrary order. To remove this ambiguity, `sort_window` packets are buffered and provided in order.

Note that an ideal `sort_window` corresponds to the *number* of out-of-order packets. It is *not* to the maximum difference in ordering of each packet.

map_streams

alias of `xrootdlib.streams.XrdXrootdMon.StreamMapper`

Submodules

xrootdlib.streams.XrdXrootdMon.fstat module

```
class Close (client: Union[xrootdlib.streams.XrdXrootdMon.map.UserInfo, xroot-
                        dlib.streams.XrdXrootdMon.map.PathAccessInfo], lfn: bytes, stats: xroot-
                        dlib.structs.XrdXrootdMon.fstat.FileCLS)
    Bases: object
    A client closed a file

    client

    classmethod from_record (record_struct: xrootdlib.structs.XrdXrootdMon.fstat.FileCLS, stod:
                            int, map_store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore)

    lfn

    stats

class Disconnect (client: xrootdlib.streams.XrdXrootdMon.map.UserInfo)
    Bases: object
    A client disconnected from the server

    client

    classmethod from_record (record_struct: xrootdlib.structs.XrdXrootdMon.fstat.FileDSC, stod:
                            int, map_store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore)

class FstatWindow (server_info: xrootdlib.streams.XrdXrootdMon.map.ServerInfo, start: int, end:
                    int, records: List[Union[xrootdlib.streams.XrdXrootdMon.fstat.Disconnect,
                    xrootdlib.streams.XrdXrootdMon.fstat.Open, xroot-
                    dlib.streams.XrdXrootdMon.fstat.Close, xrootdlib.streams.XrdXrootdMon.fstat.Transfer]])
    Bases: object
    Sequence of Open, Close and Disconnect events in a time window

    end

    records

    server_info

    start

class Open (client: Union[xrootdlib.streams.XrdXrootdMon.map.UserInfo, xroot-
                        dlib.streams.XrdXrootdMon.map.PathAccessInfo], lfn: bytes, readwrite: bool, filesize:
                        int)
    Bases: object
    A client opened a file

    client
```

```

    filesize
    classmethod from_record (record_struct: xrootdlib.structs.XrdXrootdMon.fstat.FileOPN, stod:
                               int, map_store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore)
    lfn
    readwrite

class Transfer (client: Union[xrootdlib.streams.XrdXrootdMon.map.UserInfo, xroot-
                               dlib.streams.XrdXrootdMon.map.PathAccessInfo], lfn: bytes, stats: xroot-
                               dlib.structs.XrdXrootdMon.fstat.FileXFR)
    Bases: object
    A client transferred a file
    client
    classmethod from_record (record_struct: xrootdlib.structs.XrdXrootdMon.fstat.FileXFR, stod:
                               int, map_store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore)
    lfn
    stats

digest_packet (stod: int, fstat_struct: xrootdlib.structs.XrdXrootdMon.Fstat, map_store: xroot-
                dlib.streams.XrdXrootdMon.map.MapInfoStore)
    Digest a packet containing fstat data

```

xrootdlib.streams.XrdXrootdMon.map module

```

exception MapInfoError
    Bases: Exception
    An item was not in the map store

class MapInfoStore
    Bases: object

    digest_map (stod: int, map_struct: xrootdlib.structs.XrdXrootdMon.Map)
        Digest map data from a packet

    free_path (stod: int, dictid: int) → None

    free_user (stod: int, dictid: int) → None

    get_path (stod: int, dictid: int) → xrootdlib.streams.XrdXrootdMon.map.PathAccessInfo

    get_server (stod: int, sid: int) → xrootdlib.streams.XrdXrootdMon.map.ServerInfo

    get_user (stod: int, dictid: int) → xrootdlib.streams.XrdXrootdMon.map.UserInfo

class MapInfoStoreCleaner (store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore,
                             clean_delay: int = 30)
    Bases: threading.Thread

    add_deletion (instruction, *args, **kwargs)

    run ()
        Method representing the thread's activity.

        You may override this method in a subclass. The standard run() method invokes the callable object passed
        to the object's constructor as the target argument, if any, with sequential and keyword arguments taken
        from the args and kwargs arguments, respectively.

```

```
class PathAccessInfo (user_id: xrootdlib.structs.XrdXrootdMon.map.UserId, server_info:  
                      xrootdlib.streams.XrdXrootdMon.map.ServerInfo, path_info: xroot-  
                      dlib.structs.XrdXrootdMon.map.Path)  
  
    Bases: object  
    auth_info = None  
    host  
    path  
    pid  
    protocol  
    server  
    user  
  
class ServerInfo (user_id: xrootdlib.structs.XrdXrootdMon.map.UserId, server_info: xroot-  
                   dlib.structs.XrdXrootdMon.map.SrvInfo)  
  
    Bases: object  
    host  
    instance  
    pid  
    port  
    program  
    protocol  
    sid  
    site  
    user  
    version  
  
class UserInfo (user_id: xrootdlib.structs.XrdXrootdMon.map.UserId, server_info:  
                  xrootdlib.streams.XrdXrootdMon.map.ServerInfo, auth_info: xroot-  
                  dlib.structs.XrdXrootdMon.map.AuthInfo = None)  
  
    Bases: object  
    auth_info  
    host  
    pid  
    protocol  
    server  
    user
```

xrootdlib.streams.XrdXrootdMon.redir module

```
class CmsdRedir (action: xrootdlib.structs.XrdXrootdMon.redir.XROOTD_MON, target: bytes, port: int,  
                  client: xrootdlib.streams.XrdXrootdMon.map.UserInfo, path: bytes)  
    Bases: xrootdlib.streams.XrdXrootdMon.redir.Redirection
```



```

class RedirWindow(server_info: xrootdlib.streams.XrdXrootdMon.map.ServerInfo, start: int, end: int,
                  records: List[xrootdlib.streams.XrdXrootdMon.redir.Redirection])
    Bases: object
    Sequence of events in a time window
    end
    records
    server_info
    start

class Redirection(action: xrootdlib.structs.XrdXrootdMon.redir.XROOTD_MON, target: bytes, port:
                  int, client: xrootdlib.streams.XrdXrootdMon.map.UserInfo, path: bytes)
    Bases: object
    action
    client
    classmethod from_record(record_struct: xrootdlib.structs.XrdXrootdMon.redir.Redirect, stod:
                           int, map_store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore)
    path
    port
    target

class XrootdRedir(action: xrootdlib.structs.XrdXrootdMon.redir.XROOTD_MON, target: bytes, port:
                  int, client: xrootdlib.streams.XrdXrootdMon.map.UserInfo, path: bytes)
    Bases: xrootdlib.streams.XrdXrootdMon.redir.Redirection

digest_packet(stod: int, burr_struct: xrootdlib.structs.XrdXrootdMon.Burr, map_store: xroot-
               dlib.streams.XrdXrootdMon.map.MapInfoStore)
    Digest a packet containing redir data

```

xrootdlib.streams.XrdXrootdMon.trace module

```

class Close(client: Union[xrootdlib.streams.XrdXrootdMon.map.UserInfo, xroot-
             dlib.streams.XrdXrootdMon.map.PathAccessInfo], lfn: bytes, rtot: int, wtot: int)
    Bases: object
    A client closed a file
    client
    classmethod from_record(record_struct: xrootdlib.structs.XrdXrootdMon.trace.Close, stod: int,
                           map_store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore)
    lfn
    rtot
    wtot

class Disconnect(client: xrootdlib.streams.XrdXrootdMon.map.UserInfo, duration: int, forced: bool)
    Bases: object
    A client disconnected from the server
    client
    duration

```

```
forced
classmethod from_record(record_struct: xrootdlib.structs.XrdXrootdMon.trace.Disc, stod: int,
                        map_store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore)

class Open(client: Union[xrootdlib.streams.XrdXrootdMon.map.UserInfo, xroot-
                        dlib.streams.XrdXrootdMon.map.PathAccessInfo], lfn: bytes, filesize: int)
Bases: object
A client opened a file
client
filesize
classmethod from_record(record_struct: xrootdlib.structs.XrdXrootdMon.trace.Open, stod: int,
                        map_store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore)

lfn

class ReadVector(client: Union[xrootdlib.streams.XrdXrootdMon.map.UserInfo, xroot-
                             dlib.streams.XrdXrootdMon.map.PathAccessInfo], lfn: bytes, reads: List[int])
Bases: object
client
lfn
reads

class ReadWrite(client: Union[xrootdlib.streams.XrdXrootdMon.map.UserInfo, xroot-
                             dlib.streams.XrdXrootdMon.map.PathAccessInfo], lfn: bytes, offset: int, read:
                             int, write: int)
Bases: object
A client read from or wrote to a file
client
classmethod from_record(record_struct: xrootdlib.structs.XrdXrootdMon.trace.ReadWrite,
                        stod: int, map_store: xrootdlib.streams.XrdXrootdMon.map.MapInfoStore)

lfn
offset
read
write

class TraceWindow(server_info: xrootdlib.streams.XrdXrootdMon.map.ServerInfo, start: int, end: int,
                  records: List[T])
Bases: object
Sequence of events in a time window
end
records
server_info
start

digest_packet(stod: int, buff_struct: xrootdlib.structs.XrdXrootdMon.Buff, map_store: xroot-
              dlib.streams.XrdXrootdMon.map.MapInfoStore)
Digest a packet containing trace data
```

ignore_not_implemented (*record_struct, stod, map_store*)
Skip structs that are currently not implemented

xrootdlib.streams.XrdXrootdMon.utility module

class PSeq (*pseq: int*)
Bases: `object`

Sortable *Packet Sequence*, an Integer from a wrapping (0, 255) range

Parameters *pseq* – the pseq of a packet

This represents the XRootD Packet Sequence for the purpose of comparisons. It ensures that comparisons respects wrapping from 255 to 0. In effect, a high-valued PSeq compares *less than* a low-valued PSeq.

Warning This class does not implement a full Integer interface.

exception PacketBufferExhausted
Bases: `Exception`

The buffer of packet data is exhausted

packet_from_buffer (*packet_source: IO[bytes]*)
Read a packet from a bytes buffer

xrootdlib.structs package

Representation of XRootD structs

Subpackages

xrootdlib.structs.XrdXrootdMon package

Structs used for the *Detailed Monitoring Data Format* streams sent by servers. See the `all.monitor` directive and [XRootD Monitoring](#) for details.

All types implement a `Type[T].from_XXX(buffer: bytes) -> T` constructor method, where XXX describes the appropriate section. These section are `buffer` or `record`, which represent the stream buffer either at the start of a packet or the start of a record. Unless you explicitly have a need otherwise, use `Packet.from_buffer()` to read an entire packet at a time.

class Buff (*records: List[Union[xrootdlib.structs.XrdXrootdMon.trace.AppId, xrootdlib.structs.XrdXrootdMon.trace.Close, xrootdlib.structs.XrdXrootdMon.trace.Disc, xrootdlib.structs.XrdXrootdMon.trace.Open, xrootdlib.structs.XrdXrootdMon.trace.ReadWrite, xrootdlib.structs.XrdXrootdMon.trace.ReadU, xrootdlib.structs.XrdXrootdMon.trace.ReadV, xrootdlib.structs.XrdXrootdMon.trace.Window]]*)
Bases: `object`

XrdXrootdMonBuff (“t-stream”) describing trace events

Parameters *records* – individual file operation events

The *records* field contains various trace records framed by window marks (*Window*). In other words, *records* is a *flat* sequence of one *or more* sequences of records, with marks at the start, end and between sequences.

```
classmethod from_record(record_data: bytes, record_code: bytes = b't') → xroot-
                        dlib.structs.XrdXrootdMon.Buff
Extract the record from the record portion of a stream packet buffer
```

Parameters

- **record_data** – buffer at the start of the record of a monitor stream packet
- **record_code** – the `code` field for this packet

Note The only valid record code for this class is `b't'`.

```
payload_dispath = {<XROOTD_MON.OPEN: 128>: <class 'xrootdlib.structs.XrdXrootdMon.trans
records
```

```
class Burr(sid: xrootdlib.structs.XrdXrootdMon.redir.ServerIdent, records:
              List[Union[xrootdlib.structs.XrdXrootdMon.redir.Redirect, xroot-
                          dlib.structs.XrdXrootdMon.redir.ServerIdent, xrootdlib.structs.XrdXrootdMon.redir.WindowMark]])
Bases: object
```

XrdXrootdMonBurr (“r-stream”) describing redirection events

Parameters

- **sid** – identification of the server sending events
- **records** – individual operations requested by clients

The `records` field contains redirection records (`Redirect`) framed by window marks (`WindowMark`). In other words, `records` is a *flat* sequence of one *or more* sequences of records, with marks at the start, end and between sequences.

end

```
classmethod from_record(record_data: bytes, record_code: bytes = b'r') → xroot-
                        dlib.structs.XrdXrootdMon.Burr
Extract the record from the record portion of a stream packet buffer
```

Parameters

- **record_data** – buffer at the start of the record of a monitor stream packet
- **record_code** – the `code` field for this packet

Note The only valid record code for this class is `b'r'`.

records

sid

start

```
class Fstat(tod: xrootdlib.structs.XrdXrootdMon.fstat.FileTOD, records:
               List[Union[xrootdlib.structs.XrdXrootdMon.fstat.FileTOD, xroot-
                           dlib.structs.XrdXrootdMon.fstat.FileDSC, xrootdlib.structs.XrdXrootdMon.fstat.FileOPN,
                           xrootdlib.structs.XrdXrootdMon.fstat.FileCLS, xrootdlib.structs.XrdXrootdMon.fstat.FileXFR]])
Bases: object
```

XrdXrootdMonFstat (“f-stream”) describing the general file access

Parameters

- **tod** – identifier for the server and time window
- **records** – file operations and statistics

The `records` is a flat sequence of open records (*FileOPN*), transfer records (*FileXFR*, if `fstat xfr` is configured), close records (*FileCLS*), and disconnect records (*FileDSC*). As per the specification, for every access the records are provided in this order. However, records for one access may be spread over multiple time windows.

end

classmethod `from_record`(*record_data*: bytes, *record_code*: bytes = b'f') → xrootdlib.structs.XrdXrootdMon.Fstat
Extract the record from the record portion of a stream packet buffer

Parameters

- **record_data** – buffer at the start of the record of a monitor stream packet
- **record_code** – the *code* field for this packet

Note The only valid record code for this class is b'f'.

`payload_dispath = {<recType.isClose: 0>: <class 'xrootdlib.structs.XrdXrootdMon.fstat`

`records`

`start`

`tod`

class `Header`(*code*: bytes, *pseq*: int, *plen*: int, *stod*: int)

Bases: `object`

XrdXrootdMonHeader shared by all packets

Parameters

- **code** – identifier for the record type
- **pseq** – wrapping counter for packet sequence
- **plen** – size of the packet in bytes
- **stod** – daemon start timestamp

`code`

classmethod `from_buffer`(*buffer*: bytes) → xrootdlib.structs.XrdXrootdMon.Header
Extract the header from the start of a stream packet buffer

Parameters **buffer** – buffer containing a monitor stream packet

`plen`

`pseq`

`size = 8`

`stod`

`struct_parser = <Struct object>`

class `Map`(*dictid*: int, *userid*: xrootdlib.structs.XrdXrootdMon.map.UserId, *payload*: Union[xrootdlib.structs.XrdXrootdMon.map.SrvInfo, xrootdlib.structs.XrdXrootdMon.map.Path, xrootdlib.structs.XrdXrootdMon.map.AppInfo, xrootdlib.structs.XrdXrootdMon.map.PrgInfo, xrootdlib.structs.XrdXrootdMon.map.AuthInfo, xrootdlib.structs.XrdXrootdMon.map.XfrInfo])

Bases: `object`

XrdXrootdMonMap describing transactions and general information

Parameters

- **dictid** – identifier shared by all records referring to the same information
- **userid** – identifier for the client session being monitored
- **payload** – the actual information of this message

The *Map* provides general information that applies across several monitoring events. Events of other streams reference this with the *dictid*, or the *sid* of the *UserId* of *SrvInfo* payloads. Note that in case of *SrvInfo* payloads, the *userid* contains the *server* user data.

dictid

classmethod from_record (*record_data*: bytes, *record_code*: bytes) → xrootdlib.structs.XrdXrootdMon.Map

Extract the record from the record portion of a stream packet buffer

Parameters

- **record_data** – buffer at the start of the record of a monitor stream packet
- **record_code** – the *code* field for this packet

payload

userid

class Packet (*header*: xrootdlib.structs.XrdXrootdMon.Header, *record*: Union[xrootdlib.structs.XrdXrootdMon.Map, xrootdlib.structs.XrdXrootdMon.Burr, xrootdlib.structs.XrdXrootdMon.Fstat, xrootdlib.structs.XrdXrootdMon.Buff])

Bases: object

XrdXrootdMon packet for a map, r, t or f stream

Parameters

- **header** – the header specifying type, ordering and size of the packet
- **record** – the actual information carried by the packet

classmethod from_buffer (*buffer*: bytes)

Extract the entire packet from the start of a stream packet buffer

Parameters **buffer** – buffer containing a monitor stream packet

header

record

record_dispath = {b'=': <class 'xrootdlib.structs.XrdXrootdMon.Map'>, b'd': <class 'xrootdlib.structs.XrdXrootdMon.Burr'>, b'f': <class 'xrootdlib.structs.XrdXrootdMon.Fstat'>, b'b': <class 'xrootdlib.structs.XrdXrootdMon.Buff'>}

size

PacketRecord = typing.Union[xrootdlib.structs.XrdXrootdMon.Map, xrootdlib.structs.XrdXrootdMon.Burr, xrootdlib.structs.XrdXrootdMon.Fstat, xrootdlib.structs.XrdXrootdMon.Buff]
Record types in a packet

Submodules

xrootdlib.structs.XrdXrootdMon.constants module

XROOTD_MON_SIDMASK = 281474976710655

The server ID is encoded in the lower 48 bits of the first 8 bytes

xrootdlib.structs.XrdXrootdMon.fstat module

```
class FileCLS (flags: int, size: int, fileid: int, read: int, readv: int, write: int,  
               ops: Optional[xrootdlib.structs.XrdXrootdMon.fstat.StatOPS], ssq: Op-  
               tional[xrootdlib.structs.XrdXrootdMon.fstat.StatSSQ])
```

Bases: `object`

XrdXrootdMonFileCLS indicating that a client closed a file

Parameters

- **flags** – indicator for fields present and close type
- **fileid** – file identifier (see [Map](#))
- **read** – bytes read using `read()`
- **readv** – bytes read using `readv()`
- **write** – bytes written
- **ops** – file operation statistics
- **ssq** – file operation statistic deviations

fileid

flags

classmethod from_buffer (*buffer: bytes*)

ops

read

readv

size

ssq

struct_parser = <Struct object>

write

xfr

```
class FileDSC (flags: int, dictid: str)
```

Bases: `object`

XrdXrootdMonFileDSC indicating that a client disconnected from the server

Parameters

- **flags** – unused for this record type
- **dictid** – client identifier (see [Map](#))

dictid

flags

classmethod from_buffer (*buffer: bytes*)

size = 8

struct_parser = <Struct object>

```
class FileLFNView (file_opn: xrootdlib.structs.XrdXrootdMon.fstat.FileOPN)
```

```
    Bases: object
```

```
    lfn
```

```
    user
```

```
class FileOPN (flags: int, size: int, fileid: int, filesize: int, user: Optional[int] = None, lfn: Optional[bytes] = None)
```

```
    Bases: object
```

XrdXrootdMonFileOPN indicating that a client opened a file

Parameters

- **flags** – indicator for fields present and access type
- **size** – size of the struct in bytes
- **fileid** – file identifier (see [Map](#))
- **filesize** – size of the file in bytes
- **user** – client identifier (see [Map](#))
- **lfn** – the (logical) path of the file

```
    fileid
```

```
    filesize
```

```
    flags
```

```
    classmethod from_buffer (buffer: bytes)
```

```
    fsz
```

```
    lfn
```

```
    size
```

```
    struct_parser = <Struct object>
```

```
    ufn
```

```
    user
```

```
FileRecord = typing.Union[xrootdlib.structs.XrdXrootdMon.fstat.FileTOD, xrootdlib.structs.XrdXrootdMon.fstat.FileOPN]
```

Type of records in the f stream

```
class FileTOD (flags: int, records_xfr: int, records_total: int, start: int, end: int, sid: int)
```

```
    Bases: object
```

XrdXrootdMonFileTOD identifying the sender and time range

Parameters

- **flags** – indicator for fields present
- **records_xfr** – number of [FileXFR](#) records in the packet
- **records_total** – number of [FileRecord](#) records in the packet
- **start** – timestamp of the first record
- **end** – timestamp the packet was sent
- **sid** – server identifier (see [Map](#))

```
end
```



```

    flags
    classmethod from_buffer (buffer: bytes)
    records_total
    records_xfr
    sid
    size = 24
    start
    struct_parser = <Struct object>
class FileXFR (flags: int, fileid: int, read: int, readv: int, write: int)
    Bases: object
    XrdXrootdMonFileXFR indicating file transfer statistics

    Parameters
        • flags – unused
        • fileid – client identifier (see Map)
        • read – bytes read using read ()
        • readv – bytes read using readv ()
        • write – bytes written

    fileid
    flags
    classmethod from_buffer (buffer: bytes)
    read
    readv
    size = 32
    struct_parser = <Struct object>
    write
    xfr
class StatOPS (read: int, readv: int, write: int, rsmmin: int, rsmmax: int, rsegs: int, rdmin: int, rdmax: int,
                rvmin: int, rvmax: int, wrmin: int, wrmax: int)
    Bases: object
    XrdXrootdMonStatOPS describing file operation statistics

    classmethod from_buffer (buffer: bytes)

    rdmax
    rdmin
    read
    readv
    rsegs
    rsmmax

```

```
    rsmmin
    rvmax
    rvmin
    size = 48
    struct_parser = <Struct object>
    write
    wrmax
    wrmin

class StatSSQ (read: int, readv: int, rsegs: int, write: int)
    Bases: object
    XrdXrootdMonStatSSQ describing file operation statistic deviations
    classmethod from_buffer (buffer: bytes)
    read
    readv
    rsegs
    size = 32
    struct_parser = <Struct object>
    write

class StatXFRView (file_struct: Union[xrootdlib.structs.XrdXrootdMon.fstat.FileCLS, xroot-
    dlib.structs.XrdXrootdMon.fstat.FileXFR])
    Bases: object
    read
    readv
    write

class recFval
    Bases: int, enum.Enum
    Record flags for the f stream
    forced = 1
        Disconnect prior to close
    hasLFN = 1
        XrdXroodMonFileLFN present
    hasOPS = 2
        XrdXroodMonFileOPS present
    hasRW = 2
        FileRecord opened for reads & writes
    hasSID = 1
        The sID member is present
    hasSSQ = 4
        XrdXroodMonFileSSQ present (implies hasOPS)
```

```

class recType
    Bases: int, enum.Enum

    Record type identifiers for the f stream

    isClose = 0
        XrdXrootdMonFileCLS

    isDisc = 4
        XrdXrootdMonFileDSC

    isOpen = 1
        XrdXrootdMonFileOPN

    isTime = 2
        XrdXrootdMonFileTOD

    isXFR = 3
        XrdXrootdMonFileXFR

```

xrootdlib.structs.XrdXrootdMon.map module

Contents of the XrdXrootdMonMap struct

Each struct contains a `userid/npayload` info field. The `userid` is always represented by *UserId*, while the `payload` can be any of the types of `MapPayload`.

Each class is capable of parsing its respective *portion* of the info field. That is, it expects a buffer (a `bytes`, `bytesarray` or `memoryview`) starting with the `XrdXrootdMonMap` info; trailing buffer content is allowed and the buffer is never modified.

```

class AppInfo (appinfo: bytes)
    Bases: object

    */appinfo containing un-interpreted application supplied information

    appinfo

    classmethod from_buffer (buffer: bytes)

class AuthInfo (protocol: bytes, name: bytes, host: bytes, organisation: bytes, role: bytes, group: bytes,
                 m: bytes, executable: bytes, moninfo: bytes)
    Bases: object

    */authinfo describing an authenticating user

    executable

    classmethod from_buffer (buffer: bytes)

    g
    group
    h
    host
    m
    moninfo
    n
    name

```

o
organisation

p
protocol

r
role

x

y

class Path (*path: bytes*)
Bases: `object`
**/path* containing full path name of a file being opened
classmethod from_buffer (*buffer: bytes*)
path

class PrgInfo (*xfn: bytes, tod: int, sz: int, at: int, ct: int, mt: int, fn: bytes*)
Bases: `object`
**/prginfo* describing the purging of a file
at
ct
fn
classmethod from_buffer (*buffer: bytes*)
mt
sz
tod
xfn

class SrvInfo (*pgm: bytes, ver: bytes, inst: bytes, port: int, site: bytes*)
Bases: `object`
**/srvinfo* describing the xrootd instance sending reports
classmethod from_buffer (*buffer: bytes*)
inst
pgm
port
site
ver

class UserId (*prot: bytes, user: bytes, pid: int, sid: int, host: bytes*)
Bases: `object`
*userid/** describing the user performing an action
classmethod from_buffer (*buffer: bytes*)

```

    host
    pid
    prot
    sid
    user

class XfrInfo (lfn: bytes, tod: int, sz: int, tm: int, op: str, rc: int, pd: bytes)
    Bases: object
    */xfrinfo` describing the transfer of a file
    classmethod from_buffer (buffer: bytes)
    lfn
    op
    pd
    rc
    sz
    tm
    tod

```

xrootdlib.structs.XrdXrootdMon.redir module

```

Redir = typing.Union[xrootdlib.structs.XrdXrootdMon.redir.Redirect, xrootdlib.structs.XrdXrootdMon.redir.RedLocal]
    Type of records in the r stream

```

```

class Redirect (type: xrootdlib.structs.XrdXrootdMon.redir.XROOTD_MON, subtype: xrootdlib.structs.XrdXrootdMon.redir.XROOTD_MON, dent: int, port: int, dictid: int, server: bytes, path: bytes)
    Bases: object

```

XrdXrootdMonRedir representing a redirect record

Parameters

- **type** – the type of the record, i.e. REDIRECT or REDLOCAL
- **subtype** – the requested operation, i.e. CHMOD, LOCALTE, ...
- **dent** – size of the struct in bytes plus one
- **port** – port to which the client was redirected
- **dictid** – client identifier (see [Map](#))
- **server** – hostname or address of the target server
- **path** – path of the file on the target server

```

    arg0
    arg1
    dent
    dictid
    classmethod from_buffer (buffer: bytes)

```

```
local
    Whether the redirection target is the same server

path
port
server
size
struct_parser = <Struct object>
subtype
type

class RedirectArg0View (redirect: xrootdlib.structs.XrdXrootdMon.redir.Redirect)
    Bases: object

    dent
    port
    type

class RedirectArg1View (redirect: xrootdlib.structs.XrdXrootdMon.redir.Redirect)
    Bases: object

    dictid
    path
    server
    serverpath

class ServerIdent (sid: int)
    Bases: object

    XrdXrootdMonRedir representing a server identification record

    Parameters sid – server identifier (see Map)

    classmethod from_buffer (buffer: bytes)

    sid
    size = 8
    struct_parser = <Struct object>
    type = 240
```

```
class WindowMark (timestamp: int, prev_duration: int)
    Bases: object

    XrdXrootdMonRedir representing a window timing mark
```

Parameters

- **timestamp** – timestamp when the *current* window started
- **prev_duration** – duration of the *previous* window

Note that windows are usually not adjacent: the [WindowMark](#) signifies the beginning of a *new* window. The previous window may have been closed an arbitrary time before.

To get the time range covered by a window, the `prev_duration` of the next window is required:

```
# window => tuple(start, end)
window_range[i] = (
    window_marks[i].timestamp,
    window_marks[i].timestamp + window_marks[i+1].prev_duration
)
```

arg0**arg1****classmethod from_buffer** (*buffer: bytes*)**prev_duration****size** = 8**struct_parser** = <Struct object>**timestamp****type** = 0**class WindowMarkArg0View** (*window: xrootdlib.structs.XrdXrootdMon.redir.WindowMark*)Bases: *object***type****window****class WindowMarkArg1View** (*window: xrootdlib.structs.XrdXrootdMon.redir.WindowMark*)Bases: *object***window****class XROOTD_MON**Bases: *int, enum.Enum*

XROOTD_MON_XYZ constants for the r-stream

CHMOD = 1

Change file mode

LOCATE = 2

Locate file or directory

MKDIR = 7

Create a directory or path

MV = 8

Rename a file or directory

OPENC = 4

Open file for creation

OPENDIR = 3

Open director for reading

OPENR = 5

Open file for reading

OPENW = 6

Open file for writing

PREP = 9

Prepare request

QUERY = 10
Query information request

REDIRECT = 128
Redirect event generated by cmsd

REDLOCAL = 144
Redirect event generated by xrootd

REDSID = 240
Server identification

REDTIME = 0
Window timing mark

RM = 11
Remove a file

RMDIR = 12
Remove a directory

STAT = 13
Stat a file or directory

TRUNC = 14
Truncate a file

xrootdlib.structs.XrdXrootdMon.trace module

```
class AppId(appid: bytes)
    Bases: object
    appid
    classmethod from_buffer(buffer: bytes)
    size = 16
    struct_parser = <Struct object>

class Close(rtot: int, wtot: int, dictid: int)
    Bases: object
    dictid
    classmethod from_buffer(buffer: bytes)
    rtot
    size = 16
    struct_parser = <Struct object>
    wtot

class Disc(flags: int, buflen: int, dictid: int)
    Bases: object
    buflen
    dictid
    flags
    classmethod from_buffer(buffer: bytes)
```



```

    size = 16
    struct_parser = <Struct object>
class Open (filesize: int, dictid: int)
    Bases: object
    dictid
    filesize
    classmethod from_buffer (buffer: bytes)
    size = 16
    struct_parser = <Struct object>
class Read (readid: int, count: int, buflen: int, dictid: int)
    Bases: object
    buflen
    count
    dictid
    classmethod from_buffer (buffer: bytes)
    readid
    size = 16
    struct_parser = <Struct object>
class ReadU (readid: int, count: int, buflen: int, dictid: int)
    Bases: xrootdlib.structs.XrdXrootdMon.trace.Read
class ReadV (readid: int, count: int, buflen: int, dictid: int)
    Bases: xrootdlib.structs.XrdXrootdMon.trace.Read
class ReadWrite (val: int, buflen: int, dictid: int)
    Bases: object
    buflen
    dictid
    classmethod from_buffer (buffer: bytes)
    readlen
    size = 16
    struct_parser = <Struct object>
    val
    writelen
TRACE_SIZE = 16
    Entries in the I/O and non-I/O event streams are always of fixed size (i.e., 16 characters)
class Window (sid: int, end: int, start: int)
    Bases: object
    end
    classmethod from_buffer (buffer: bytes)

```

```
sid
size = 16
start
struct_parser = <Struct object>
class XROOTD_MON
    Bases: int, enum.Enum
    An enumeration.
    APPID = 160
        Application provided marker
    BOUNDP = 2
        Entry for a bound path
    CLOSE = 192
        File has been closed
    DISC = 208
        Client has disconnected
    FORCED = 1
        Entry due to forced disconnect
    OPEN = 128
        File has been opened
    READU = 145
        Unpacked details for kXR_readv
    READV = 144
        Details for a kXR_readv request
    WINDOW = 224
        Window timing mark
```

1.1.2 Submodules

xrootdlib.utility module

parse_cgi (*cgi_data: AnyStr*) → Dict[AnyStr, AnyStr]
Parse cgi data in the form &key1=value1&key2=value2 to a mapping

Parameters *cgi_data* – raw CGI data in as unicode or bytes

Returns a mapping from keys to values

Note that this does not perform any implicit type conversions: keys and values have the same type as *cgi_data*. For example, a value of `b'1'` is not converted to the integer 1.

```
>>> parse_cgi(b'&foo=1')
{b'foo': b'1'}
```

slot_repr (*instance*)

The `xrootdlib` offers building blocks and basic tools to work with the [XRootD](#) data access middleware. It is meant to facilitate auxiliary work, such as monitoring, accounting and orchestration.

Contributing and Feedback

The project is hosted on [github](#). If you have issues or suggestion, check the issue tracker: For direct contributions, feel free to fork the [development branch](#) and open a pull request.

2.1 Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)

Documentation built from xrootdlib 0.2.0 at Sep 23, 2021.

X

- [xrootdlib](#), [1](#)
- [xrootdlib.streams](#), [1](#)
- [xrootdlib.streams.XrdXrootdMon](#), [1](#)
- [xrootdlib.streams.XrdXrootdMon.fstat](#), [2](#)
- [xrootdlib.streams.XrdXrootdMon.map](#), [3](#)
- [xrootdlib.streams.XrdXrootdMon.redir](#), [4](#)
- [xrootdlib.streams.XrdXrootdMon.trace](#), [5](#)
- [xrootdlib.streams.XrdXrootdMon.utility](#),
[7](#)
- [xrootdlib.structs](#), [7](#)
- [xrootdlib.structs.XrdXrootdMon](#), [7](#)
- [xrootdlib.structs.XrdXrootdMon.constants](#),
[10](#)
- [xrootdlib.structs.XrdXrootdMon.fstat](#),
[11](#)
- [xrootdlib.structs.XrdXrootdMon.map](#), [15](#)
- [xrootdlib.structs.XrdXrootdMon.redir](#),
[17](#)
- [xrootdlib.structs.XrdXrootdMon.trace](#),
[20](#)
- [xrootdlib.utility](#), [22](#)

A

action (*Redirection attribute*), 5
 add_deletion() (*MapInfoStoreCleaner method*), 3
 appid (*AppId attribute*), 20
 AppId (*class in xrootdlib.structs.XrdXrootdMon.trace*), 20
 APPID (*XROOTD_MON attribute*), 22
 appinfo (*AppInfo attribute*), 15
 AppInfo (*class in xrootdlib.structs.XrdXrootdMon.map*), 15
 arg0 (*Redirect attribute*), 17
 arg0 (*WindowMark attribute*), 19
 arg1 (*Redirect attribute*), 17
 arg1 (*WindowMark attribute*), 19
 at (*PrgInfo attribute*), 16
 auth_info (*PathAccessInfo attribute*), 4
 auth_info (*UserInfo attribute*), 4
 AuthInfo (*class in xrootdlib.structs.XrdXrootdMon.map*), 15

B

BOUNDP (*XROOTD_MON attribute*), 22
 Buff (*class in xrootdlib.structs.XrdXrootdMon*), 7
 buflen (*Disc attribute*), 20
 buflen (*Read attribute*), 21
 buflen (*ReadWrite attribute*), 21
 Burr (*class in xrootdlib.structs.XrdXrootdMon*), 8

C

CHMOD (*XROOTD_MON attribute*), 19
 client (*Close attribute*), 2, 5
 client (*Disconnect attribute*), 2, 5
 client (*Open attribute*), 2, 6
 client (*ReadVector attribute*), 6
 client (*ReadWrite attribute*), 6
 client (*Redirection attribute*), 5
 client (*Transfer attribute*), 3
 Close (*class in xrootdlib.streams.XrdXrootdMon.fstat*), 2

Close (*class in xrootdlib.streams.XrdXrootdMon.trace*), 5
 Close (*class in xrootdlib.structs.XrdXrootdMon.trace*), 20
 CLOSE (*XROOTD_MON attribute*), 22
 CmsdRedir (*class in xrootdlib.streams.XrdXrootdMon.redir*), 4
 code (*Header attribute*), 9
 count (*Read attribute*), 21
 ct (*PrgInfo attribute*), 16

D

dent (*Redirect attribute*), 17
 dent (*RedirectArg0View attribute*), 18
 dictid (*Close attribute*), 20
 dictid (*Disc attribute*), 20
 dictid (*FileDSC attribute*), 11
 dictid (*Map attribute*), 10
 dictid (*Open attribute*), 21
 dictid (*Read attribute*), 21
 dictid (*ReadWrite attribute*), 21
 dictid (*Redirect attribute*), 17
 dictid (*RedirectArg1View attribute*), 18
 digest_map() (*MapInfoStore method*), 3
 digest_packet() (*in module xrootdlib.streams.XrdXrootdMon.fstat*), 3
 digest_packet() (*in module xrootdlib.streams.XrdXrootdMon.redir*), 5
 digest_packet() (*in module xrootdlib.streams.XrdXrootdMon.trace*), 6
 Disc (*class in xrootdlib.structs.XrdXrootdMon.trace*), 20
 DISC (*XROOTD_MON attribute*), 22
 Disconnect (*class in xrootdlib.streams.XrdXrootdMon.fstat*), 2
 Disconnect (*class in xrootdlib.streams.XrdXrootdMon.trace*), 5
 duration (*Disconnect attribute*), 5

E

end (*Burr attribute*), 8

end (*FileTOD attribute*), 12
end (*Fstat attribute*), 9
end (*FstatWindow attribute*), 2
end (*RedirWindow attribute*), 5
end (*TraceWindow attribute*), 6
end (*Window attribute*), 21
executable (*AuthInfo attribute*), 15

F

FileCLS (class in xroot-
dlib.structs.XrdXrootdMon.fstat), 11
FileDSC (class in xroot-
dlib.structs.XrdXrootdMon.fstat), 11
fileid (*FileCLS attribute*), 11
fileid (*FileOPN attribute*), 12
fileid (*FileXFR attribute*), 13
FileLFNView (class in xroot-
dlib.structs.XrdXrootdMon.fstat), 11
FileOPN (class in xroot-
dlib.structs.XrdXrootdMon.fstat), 12
FileRecord (in module xroot-
dlib.structs.XrdXrootdMon.fstat), 12
filesize (*FileOPN attribute*), 12
filesize (*Open attribute*), 2, 6, 21
FileTOD (class in xroot-
dlib.structs.XrdXrootdMon.fstat), 12
FileXFR (class in xroot-
dlib.structs.XrdXrootdMon.fstat), 13
flags (*Disc attribute*), 20
flags (*FileCLS attribute*), 11
flags (*FileDSC attribute*), 11
flags (*FileOPN attribute*), 12
flags (*FileTOD attribute*), 12
flags (*FileXFR attribute*), 13
fn (*PrgInfo attribute*), 16
forced (*Disconnect attribute*), 5
forced (*recFval attribute*), 14
FORCED (*XROOTD_MON attribute*), 22
free_path() (*MapInfoStore method*), 3
free_user() (*MapInfoStore method*), 3
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.fstat.FileCLS
class method), 11
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.fstat.FileDSC
class method), 11
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.fstat.FileOPN
class method), 12
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.fstat.FileTOD
class method), 13
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.fstat.FileXFR

class method), 13
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.fstat.StatOPS
class method), 13
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.fstat.StatSSQ
class method), 14
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.Header class
method), 9
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.map.AppInfo
class method), 15
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.map.AuthInfo
class method), 15
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.map.Path class
method), 16
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.map.PrgInfo class
method), 16
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.map.SrvInfo class
method), 16
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.map.UserId class
method), 16
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.map.XfrInfo class
method), 17
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.Packet class
method), 10
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.redir.Redirect
class method), 17
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.redir.ServerIdent
class method), 18
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.redir.WindowMark
class method), 19
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.trace.AppId class
method), 20
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.trace.Close class
method), 20
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.trace.Disc class
method), 20
from_buffer() (xroot-
dlib.structs.XrdXrootdMon.trace.Open class

method), 21

from_buffer() (xrootdlib.structs.XrdXrootdMon.trace.Read class method), 21

from_buffer() (xrootdlib.structs.XrdXrootdMon.trace.ReadWrite class method), 21

from_buffer() (xrootdlib.structs.XrdXrootdMon.trace.Window class method), 21

from_record() (xrootdlib.streams.XrdXrootdMon.fstat.Close class method), 2

from_record() (xrootdlib.streams.XrdXrootdMon.fstat.Disconnect class method), 2

from_record() (xrootdlib.streams.XrdXrootdMon.fstat.Open class method), 3

from_record() (xrootdlib.streams.XrdXrootdMon.fstat.Transfer class method), 3

from_record() (xrootdlib.streams.XrdXrootdMon.redir.Redirection class method), 5

from_record() (xrootdlib.streams.XrdXrootdMon.trace.Close class method), 5

from_record() (xrootdlib.streams.XrdXrootdMon.trace.Disconnect class method), 6

from_record() (xrootdlib.streams.XrdXrootdMon.trace.Open class method), 6

from_record() (xrootdlib.streams.XrdXrootdMon.trace.ReadWrite class method), 6

from_record() (xrootdlib.structs.XrdXrootdMon.Buff class method), 7

from_record() (xrootdlib.structs.XrdXrootdMon.Burr class method), 8

from_record() (xrootdlib.structs.XrdXrootdMon.Fstat class method), 9

from_record() (xrootdlib.structs.XrdXrootdMon.Map class method), 10

Fstat (class in xrootdlib.structs.XrdXrootdMon), 8

FstatWindow (class in xrootdlib.streams.XrdXrootdMon.fstat), 2

fsz (FileOPN attribute), 12

G

g (AuthInfo attribute), 15

get_path() (MapInfoStore method), 3

get_server() (MapInfoStore method), 3

get_user() (MapInfoStore method), 3

group (AuthInfo attribute), 15

H

h (AuthInfo attribute), 15

hasLFN (recFval attribute), 14

hasOPS (recFval attribute), 14

hasRW (recFval attribute), 14

hasSID (recFval attribute), 14

hasSSQ (recFval attribute), 14

Header (class in xrootdlib.structs.XrdXrootdMon), 9

header (Packet attribute), 10

host (AuthInfo attribute), 15

host (PathAccessInfo attribute), 4

host (ServerInfo attribute), 4

host (UserId attribute), 16

host (UserInfo attribute), 4

I

ignore_not_implemented() (in module xrootdlib.streams.XrdXrootdMon.trace), 6

inst (SrvInfo attribute), 16

instance (ServerInfo attribute), 4

isClose (recType attribute), 15

isDisc (recType attribute), 15

isOpen (recType attribute), 15

isTime (recType attribute), 15

isXFR (recType attribute), 15

L

lfn (Close attribute), 2, 5

lfn (FileLFNView attribute), 12

lfn (FileOPN attribute), 12

lfn (Open attribute), 3, 6

lfn (ReadVector attribute), 6

lfn (ReadWrite attribute), 6

lfn (Transfer attribute), 3

lfn (XfrInfo attribute), 17

local (Redirect attribute), 18

LOCATE (XROOTD_MON attribute), 19

M

m (AuthInfo attribute), 15

Map (class in xrootdlib.structs.XrdXrootdMon), 9

map_streams (in module xrootdlib.streams.XrdXrootdMon), 2

MapInfoError, 3

MapInfoStore (class in xrootdlib.streams.XrdXrootdMon.map), 3

MapInfoStoreCleaner (class in
dlib.streams.XrdXrootdMon.map), 3
MKDIR (XROOTD_MON attribute), 19
moninfo (AuthInfo attribute), 15
mt (PrgInfo attribute), 16
MV (XROOTD_MON attribute), 19

N

n (AuthInfo attribute), 15
name (AuthInfo attribute), 15

O

o (AuthInfo attribute), 15
offset (ReadWrite attribute), 6
op (XfrInfo attribute), 17
Open (class in xrootdlib.streams.XrdXrootdMon.fstat), 2
Open (class in xrootdlib.streams.XrdXrootdMon.trace), 6
Open (class in xrootdlib.structs.XrdXrootdMon.trace), 21
OPEN (XROOTD_MON attribute), 22
OPENC (XROOTD_MON attribute), 19
OPENDIR (XROOTD_MON attribute), 19
OPENR (XROOTD_MON attribute), 19
OPENW (XROOTD_MON attribute), 19
ops (FileCLS attribute), 11
organisation (AuthInfo attribute), 16

P

p (AuthInfo attribute), 16
Packet (class in xrootdlib.structs.XrdXrootdMon), 10
packet_from_buffer() (in module xroot-
dlib.streams.XrdXrootdMon.utility), 7
PacketBufferExhausted, 7
PacketRecord (in module xroot-
dlib.structs.XrdXrootdMon), 10
parse CGI() (in module xrootdlib.utility), 22
Path (class in xrootdlib.structs.XrdXrootdMon.map), 16
path (Path attribute), 16
path (PathAccessInfo attribute), 4
path (Redirect attribute), 18
path (RedirectArgView attribute), 18
path (Redirection attribute), 5
PathAccessInfo (class in xroot-
dlib.streams.XrdXrootdMon.map), 3
payload (Map attribute), 10
payload_dispath (Buff attribute), 8
payload_dispath (Fstat attribute), 9
pd (XfrInfo attribute), 17
pgm (SrvInfo attribute), 16
pid (PathAccessInfo attribute), 4
pid (ServerInfo attribute), 4
pid (UserId attribute), 17
pid (UserInfo attribute), 4
plen (Header attribute), 9
port (Redirect attribute), 18

port (RedirectArgView attribute), 18
port (Redirection attribute), 5
port (ServerInfo attribute), 4
port (SrvInfo attribute), 16
PREP (XROOTD_MON attribute), 19
prev_duration (WindowMark attribute), 19
PrgInfo (class in xroot-
dlib.structs.XrdXrootdMon.map), 16
program (ServerInfo attribute), 4
prot (UserId attribute), 17
protocol (AuthInfo attribute), 16
protocol (PathAccessInfo attribute), 4
protocol (ServerInfo attribute), 4
protocol (UserInfo attribute), 4
PSeq (class in xrootdlib.streams.XrdXrootdMon.utility),
7
pseq (Header attribute), 9

Q

QUERY (XROOTD_MON attribute), 19

R

r (AuthInfo attribute), 16
rc (XfrInfo attribute), 17
rdmax (StatOPS attribute), 13
rdmin (StatOPS attribute), 13
Read (class in xrootdlib.structs.XrdXrootdMon.trace), 21
read (FileCLS attribute), 11
read (FileXFR attribute), 13
read (ReadWrite attribute), 6
read (StatOPS attribute), 13
read (StatSSQ attribute), 14
read (StatXFRView attribute), 14
readid (Read attribute), 21
readlen (ReadWrite attribute), 21
reads (ReadVector attribute), 6
ReadU (class in xrootdlib.structs.XrdXrootdMon.trace),
21
READU (XROOTD_MON attribute), 22
ReadV (class in xrootdlib.structs.XrdXrootdMon.trace),
21
readv (FileCLS attribute), 11
readv (FileXFR attribute), 13
readv (StatOPS attribute), 13
readv (StatSSQ attribute), 14
readv (StatXFRView attribute), 14
READV (XROOTD_MON attribute), 22
ReadVector (class in xroot-
dlib.streams.XrdXrootdMon.trace), 6
ReadWrite (class in xroot-
dlib.streams.XrdXrootdMon.trace), 6
ReadWrite (class in xroot-
dlib.structs.XrdXrootdMon.trace), 21
readwrite (Open attribute), 3

recFval (class in dlib.structs.XrdXrootdMon.fstat), 14
 record (Packet attribute), 10
 record_dispath (Packet attribute), 10
 records (Buff attribute), 8
 records (Burr attribute), 8
 records (Fstat attribute), 9
 records (FstatWindow attribute), 2
 records (RedirWindow attribute), 5
 records (TraceWindow attribute), 6
 records_total (FileTOD attribute), 13
 records_xfr (FileTOD attribute), 13
 recType (class in dlib.structs.XrdXrootdMon.fstat), 14
 Redir (in module dlib.structs.XrdXrootdMon.redir), 17
 Redirect (class in dlib.structs.XrdXrootdMon.redir), 17
 REDIRECT (XROOTD_MON attribute), 20
 RedirectArg0View (class in dlib.structs.XrdXrootdMon.redir), 18
 RedirectArg1View (class in dlib.structs.XrdXrootdMon.redir), 18
 Redirection (class in dlib.streams.XrdXrootdMon.redir), 5
 RedirWindow (class in dlib.streams.XrdXrootdMon.redir), 4
 REDLOCAL (XROOTD_MON attribute), 20
 REDSID (XROOTD_MON attribute), 20
 REDTIME (XROOTD_MON attribute), 20
 RM (XROOTD_MON attribute), 20
 RMDIR (XROOTD_MON attribute), 20
 role (AuthInfo attribute), 16
 rsegs (StatOPS attribute), 13
 rsegs (StatSSQ attribute), 14
 rsmax (StatOPS attribute), 13
 rsmin (StatOPS attribute), 13
 rtot (Close attribute), 5, 20
 run () (MapInfoStoreCleaner method), 3
 rvmax (StatOPS attribute), 14
 rvmin (StatOPS attribute), 14

S

server (PathAccessInfo attribute), 4
 server (Redirect attribute), 18
 server (RedirectArg1View attribute), 18
 server (UserInfo attribute), 4
 server_info (FstatWindow attribute), 2
 server_info (RedirWindow attribute), 5
 server_info (TraceWindow attribute), 6
 ServerIdent (class in dlib.structs.XrdXrootdMon.redir), 18
 ServerInfo (class in dlib.streams.XrdXrootdMon.map), 4
 xroot- serverpath (RedirectArg1View attribute), 18
 sid (Burr attribute), 8
 sid (FileTOD attribute), 13
 sid (ServerIdent attribute), 18
 sid (ServerInfo attribute), 4
 sid (UserId attribute), 17
 sid (Window attribute), 21
 site (ServerInfo attribute), 4
 site (SrvInfo attribute), 16
 size (AppId attribute), 20
 size (Close attribute), 20
 size (Disc attribute), 20
 xroot- size (FileCLS attribute), 11
 size (FileDSC attribute), 11
 xroot- size (FileOPN attribute), 12
 size (FileTOD attribute), 13
 xroot- size (FileXFR attribute), 13
 size (Header attribute), 9
 size (Open attribute), 21
 xroot- size (Packet attribute), 10
 size (Read attribute), 21
 xroot- size (ReadWrite attribute), 21
 size (Redirect attribute), 18
 xroot- size (ServerIdent attribute), 18
 size (StatOPS attribute), 14
 xroot- size (StatSSQ attribute), 14
 size (Window attribute), 22
 size (WindowMark attribute), 19
 slot_repr () (in module xrootdlib.utility), 22
 SrvInfo (class in xroot- dlib.structs.XrdXrootdMon.map), 16
 ssq (FileCLS attribute), 11
 start (Burr attribute), 8
 start (FileTOD attribute), 13
 start (Fstat attribute), 9
 start (FstatWindow attribute), 2
 start (RedirWindow attribute), 5
 start (TraceWindow attribute), 6
 start (Window attribute), 22
 STAT (XROOTD_MON attribute), 20
 StatOPS (class in xroot- dlib.structs.XrdXrootdMon.fstat), 13
 stats (Close attribute), 2
 stats (Transfer attribute), 3
 StatSSQ (class in xroot- dlib.structs.XrdXrootdMon.fstat), 14
 StatXFRView (class in xroot- dlib.structs.XrdXrootdMon.fstat), 14
 stod (Header attribute), 9
 stream_packets (class in xroot- dlib.streams.XrdXrootdMon), 1
 struct_parser (AppId attribute), 20
 xroot- struct_parser (Close attribute), 20
 struct_parser (Disc attribute), 21

struct_parser (*FileCLS attribute*), 11
 struct_parser (*FileDSC attribute*), 11
 struct_parser (*FileOPN attribute*), 12
 struct_parser (*FileTOD attribute*), 13
 struct_parser (*FileXFR attribute*), 13
 struct_parser (*Header attribute*), 9
 struct_parser (*Open attribute*), 21
 struct_parser (*Read attribute*), 21
 struct_parser (*ReadWrite attribute*), 21
 struct_parser (*Redirect attribute*), 18
 struct_parser (*ServerIdent attribute*), 18
 struct_parser (*StatOPS attribute*), 14
 struct_parser (*StatSSQ attribute*), 14
 struct_parser (*Window attribute*), 22
 struct_parser (*WindowMark attribute*), 19
 subtype (*Redirect attribute*), 18
 sz (*PrgInfo attribute*), 16
 sz (*XfrInfo attribute*), 17

T

target (*Redirection attribute*), 5
 timestamp (*WindowMark attribute*), 19
 tm (*XfrInfo attribute*), 17
 tod (*Fstat attribute*), 9
 tod (*PrgInfo attribute*), 16
 tod (*XfrInfo attribute*), 17
 TRACE_SIZE (in module *xrootdlib.structs.XrdXrootdMon.trace*), 21
 TraceWindow (class in *xrootdlib.streams.XrdXrootdMon.trace*), 6
 Transfer (class in *xrootdlib.streams.XrdXrootdMon.fstat*), 3
 TRUNC (*XROOTD_MON attribute*), 20
 type (*Redirect attribute*), 18
 type (*RedirectArg0View attribute*), 18
 type (*ServerIdent attribute*), 18
 type (*WindowMark attribute*), 19
 type (*WindowMarkArg0View attribute*), 19

U

ufn (*FileOPN attribute*), 12
 user (*FileLFNView attribute*), 12
 user (*FileOPN attribute*), 12
 user (*PathAccessInfo attribute*), 4
 user (*ServerInfo attribute*), 4
 user (*UserId attribute*), 17
 user (*UserInfo attribute*), 4
 UserId (class in *xrootdlib.structs.XrdXrootdMon.map*), 16
 userid (*Map attribute*), 10
 UserInfo (class in *xrootdlib.streams.XrdXrootdMon.map*), 4

V

val (*ReadWrite attribute*), 21
 ver (*SrvInfo attribute*), 16
 version (*ServerInfo attribute*), 4

W

Window (class in *xrootdlib.structs.XrdXrootdMon.trace*), 21
 window (*WindowMarkArg0View attribute*), 19
 window (*WindowMarkArg1View attribute*), 19
 WINDOW (*XROOTD_MON attribute*), 22
 WindowMark (class in *xrootdlib.structs.XrdXrootdMon.redir*), 18
 WindowMarkArg0View (class in *xrootdlib.structs.XrdXrootdMon.redir*), 19
 WindowMarkArg1View (class in *xrootdlib.structs.XrdXrootdMon.redir*), 19
 write (*FileCLS attribute*), 11
 write (*FileXFR attribute*), 13
 write (*ReadWrite attribute*), 6
 write (*StatOPS attribute*), 14
 write (*StatSSQ attribute*), 14
 write (*StatXFRView attribute*), 14
 writelen (*ReadWrite attribute*), 21
 wrmax (*StatOPS attribute*), 14
 wrmin (*StatOPS attribute*), 14
 wtot (*Close attribute*), 5, 20

X

x (*AuthInfo attribute*), 16
 xfn (*PrgInfo attribute*), 16
 xfr (*FileCLS attribute*), 11
 xfr (*FileXFR attribute*), 13
 XfrInfo (class in *xrootdlib.structs.XrdXrootdMon.map*), 17
 XROOTD_MON (class in *xrootdlib.structs.XrdXrootdMon.redir*), 19
 XROOTD_MON (class in *xrootdlib.structs.XrdXrootdMon.trace*), 22
 XROOTD_MON_SIDMASK (in module *xrootdlib.structs.XrdXrootdMon.constants*), 10
 xrootdlib (module), 1
 xrootdlib.streams (module), 1
 xrootdlib.streams.XrdXrootdMon (module), 1
 xrootdlib.streams.XrdXrootdMon.fstat (module), 2
 xrootdlib.streams.XrdXrootdMon.map (module), 3
 xrootdlib.streams.XrdXrootdMon.redir (module), 4
 xrootdlib.streams.XrdXrootdMon.trace (module), 5
 xrootdlib.streams.XrdXrootdMon.utility (module), 7

`xrootdlib.structs` (*module*), [7](#)
`xrootdlib.structs.XrdXrootdMon` (*module*), [7](#)
`xrootdlib.structs.XrdXrootdMon.constants`
 (*module*), [10](#)
`xrootdlib.structs.XrdXrootdMon.fstat`
 (*module*), [11](#)
`xrootdlib.structs.XrdXrootdMon.map` (*mod-*
 ule), [15](#)
`xrootdlib.structs.XrdXrootdMon.redir`
 (*module*), [17](#)
`xrootdlib.structs.XrdXrootdMon.trace`
 (*module*), [20](#)
`xrootdlib.utility` (*module*), [22](#)
`XrootdRedir` (class in *xroot-*
 dlib.streams.XrdXrootdMon.redir), [5](#)

Y

`y` (*AuthInfo attribute*), [16](#)