DKB Documentation

DKB team

CONTENTS:

1	pyD]	B package	1
	1.1	Quickstart guide	1
		Subpackages	
2	Indi	s and tables	23
Ру	thon]	odule Index	25
In	dex		27

CHAPTER

ONE

PyDKB package

Common library for Data Knowledge Base Dataflow stages development.

Dataflow ETL process (extract-transform-load) for populating internal DKB storages and keeping them up to date

Dataflow stage Logical step of ETL process, implemented as standalone executable program (worker)

Dataflow stages are standalone programs, but can be combined into a pipeline by means of Kafka-based supervising program. For details about program compatibility with the supervisor please check documentation for the Metadata Integration Topology Management System (MInT MS) workers¹. Worker program can be written in any language; pyDKB is intended to simplify this process for Python.

Warning: There are three types of stages corresponding three types of ETL operations: *source connector* (data extraction), *processor* (transformation) and *sink connector* (load to internal DKB storage). Currently pyDKB library can be used only for *processor* stages, but in future versions *connector* stages will also be supported.

1.1 Quickstart guide

To create simple processor stage application first decide input and output data format. In following examples we will work with data in JSON format (for the full list of supported formats check pyDKB.dataflow.messages section of this documentation).

Now let's start writing example processor welcome.py and implement message handler – functional part of the stage (operations to be performed on data flow units):

```
from pyDKB.dataflow.messages import JSONMessage

def my_process(stage, message):
    """ Single message processing. """
    input_data = message.content()
    name = input_data.get('name')
    if name:
        out_data = {'message': "Welcome, %s!" % name}
        out_message = JSONMessage(out_data)
        stage.output(out_message)
    return True
```

Function must take two arguments: stage (stage context object) and message (input message, which should be transformed by our stage). Message is a smallest data unit in the data flow running through the processor, and every message is to be processed independently of previous or following ones. message.content() and

¹ WIP

JSONMessage (out_data) statements are used to decode/encode message to/from Python dict object. Message, passed to the function, is taken from the input data flow; to write new message(s) to the output data flow, stage.output (out_message) is used. It can be used as many times as many output messages were generated (or once with the list of messages). In our example, messages without key 'name' will produce no output messages, so stage.output() will not be called at all. In terms of data flow it means that the input message is filtered out and will not reach the *sink connector*.

Boolean return value of my_process() indicates if the processing was successful or not. If processing failed (False is returned), produced output messages will be dropped to avoid loading sketchy information into the DKB storages.

Now as we have processing logic implemented, we need to turn it into fully functional application. Add following lines to welcome.py:

First we create stage object and indicate that input and output message format is JSON: stage = JSONProcessorStage() (for full list of processors check *pyDKB.dataflow.stage package* section of this documentation); then set stage processing function to our function my_process(), parse command line arguments (stage.parse_args(sys.argv[1:])) and start the stage execution.

Easy, right?

It's time to run our application. Create input data sample input.ndjson with following lines:

```
{"name": "James", "city": "New York"}
{"user": "Jonathan", "role": "support"}
{"name": "John Smith"}
```

and type:

```
$ python welcome.py --dest s input.ndjson
{"message": "Welcome, James!"}
{"message": "Welcome, John Smith!"}
```

--dest s indicates that output destination is (s)tdout (default destination is file). For full information about modes in which the stage application can be used, run python welcome.py -h.

That's it, your first application is ready to be integrated into an ETL process as data processing node. For details about ETL process creation check *MInT Supervisor*² documentation.

² WIP

1.2 Subpackages

1.2.1 pyDKB.common package

Common modules.

Submodules

pyDKB.common.Type module

Abstract class for type definitions.

Example

```
class pyDKB.common.Type.Type(*args)
    Bases: object
```

Abstract class for type definitions.

Member names (str) are passed to the constructor as positional arguments.

```
add (name)
```

Add member.

Parameters name (str) – name of the member to be added

```
hasMember(val)
```

Check if the member exists (by value).

Parameters val (int) – member to be checked

Returns True/False

Return type bool

member (name)

Check if the member exists (by name).

Parameters name (str) – name to be checked

Returns member value or False if member does not exist

Return type int, bool

memberName(val)

Return string name of the member.

Parameters val (*int*) – member to retrieve name for

Returns member name of False if member does not exist

Return type str, bool

pyDKB.common.custom_readline module

Implementation of "readline"-like functionality for custom separator.

Todo: make import of fcntl (or of this module) optional to avoid errors when library is used under Windows.

```
pyDKB.common.custom_readline.custom_readline(f, newline)
```

Read lines with custom line separator.

Construct generator with readline-like functionality: with every call of next () method it will read data from f untill the newline separator is found; then yields what was read.

Warning: the last line can be incomplete, if the input data flow is interrupted in the middle of data writing.

Parameters

- **f** (file) readable file object
- **newline** (str) delimeter to be used instead of \n

Returns iterable object

Return type generator

Todo:

- make last "line" handling more strict: no newline == no line;
- rethink function name (as "line" is actually a "message");
- move functionality to pyDKB.dataflow.communication submodule)

pyDKB.common.exceptions module

Definition of common modules exceptions

```
exception pyDKB.common.exceptions.HDFSException
```

Bases: exceptions.RuntimeError

Base Exception for HDFS module.

pyDKB.common.hdfs module

Utils to interact with HDFS.

¹ https://github.com/PanDAWMS/dkb/pull/129

```
pyDKB.common.hdfs.File(fname)
     Get and open temporary local copy of HDFS file
     Return value: open file object (TemporaryFile).
pyDKB.common.hdfs.basename(path)
     Return file name without path.
pyDKB.common.hdfs.check_stderr(proc, timeout=None, max_lines=1)
     Wait till the end of the subprocess and send its STDERR to STDERR.
     Output only MAX_LINES of the STDERR to the current STDERR; if MAX_LINES == None, output all the
     STDERR.
     Return value is the subprocess' return code.
pyDKB.common.hdfs.dirname (path)
     Return dirname without filename.
pyDKB.common.hdfs.getfile(fname)
     Download file from HDFS.
     Return value: file name (without directory)
pyDKB.common.hdfs.join(path, filename)
     Join path and filename.
pyDKB.common.hdfs.listdir(dirname, mode='a')
     List files and/or subdirectories of HDFS directory.
     Parameters: dirname – directory to list mode – 'a': list all objects
              'f': list files 'd': list subdirectories
pyDKB.common.hdfs.makedirs(dirname)
     Try to create directory (with parents).
pyDKB.common.hdfs.movefile (fname, dest)
     Move local file to HDFS.
\verb"pyDKB.common.hdfs.putfile" (\textit{fname}, \textit{dest})
     Upload file to HDFS.
```

pyDKB.common.json_utils module

Utils to work with JSON objects.

In context of Python, JSON¹ objects may be considered as structures consisting of six types of elements:

- · dictionaries,
- lists,
- strings,
- · numbers,
- True/False,
- Null.

DKB project uses JSON for storing various information and transferring it between stages. This module contains functions which simplify some aspects of retrieving data from JSON objects.

¹ https://www.json.org/

```
pyDKB.common.json_utils.nestedKeys(key)
```

Transform string with nested keys into list.

String should contain keys separated by dot. If a key contains dot itself, the key must be put between matching quotation marks. Quotation marks inside the keys (not preceding or following a dot) are treated as ordinary symbols. If a list is given instead of str, it is returned without changes.

```
Parameters key (str, list) - nested keys
```

Returns nested keys

Return type list

```
pyDKB.common.json_utils.valueByKey(json_data, key)
```

Return value by a chain (list) of nested keys.

It is common for JSON objects to contain many layers of dictionaries nested in other dictionaries – this function extracts the data from such constructions according to given string or list with keys.

Parameters

- json_data (dict) to search in
- key (str, list) nested keys

Returns value (None if failed)

Return type depends on value, NoneType

1.2.2 pyDKB.dataflow package

Dataflow organization utils.

Subpackages

pyDKB.dataflow.communication package

```
pyDKB.dataflow.communication
```

```
pyDKB.dataflow.communication.Message (msg_type) Return class XXXMessage, where XXX is the passed type.
```

Subpackages

pyDKB.dataflow.communication.consumer package

Consumer submodule init file.

consumerClass = None

```
class pyDKB.dataflow.communication.consumer.ConsumerBuilder(config={})
    Bases: object
    Constructor for Consumer instance.
build(config={})
    Return constructed consumer.
```

```
Set data source for the consumer.
     setType(Type)
          Set message type for the consumer.
Submodules
pyDKB.dataflow.communication.consumer.Consumer module
pyDKB.dataflow.communication.consumer.Consumer
class pyDKB.dataflow.communication.consumer.Consumer.Consumer(config={/})
     Bases: object
     Data consumer implementation.
     close()
         Close opened data stream and data source.
     config = None
     get_message()
         Get new message from current source.
          Return values: Message object False (failed to parse message) None (all input sources are empty)
     get source info()
         Return current source info.
     get_stream()
         Get input stream linked to the current source.
          Return value: InputStream None (no sources left to read from)
     init_stream()
          Init input stream.
     log(message, level=3)
          Output log message with given log level.
     message_class()
         Return message class.
     message_type = None
     next()
          Return new Message, read from input stream.
     reconfigure (config={})
          (Re)initialize consumer with stage config arguments.
     reset stream()
          Reset input stream to the current source.
     set_message_type (Type)
         Set input message type.
exception pyDKB.dataflow.communication.consumer.Consumer.ConsumerException
     Bases: pyDKB.dataflow.exceptions.DataflowException
     Dataflow Consumer exception.
```

setSource (source)

pyDKB.dataflow.communication.consumer.FileConsumer module

pyDKB.dataflow.communication.consumer.FileConsumer

Data consumer implementation for common (static) files.

TODO: think about:

- · updatable files
- pipes (better, from the point of StreamConsumer)
- round-robin (for updatable sources)
- . .

```
class pyDKB.dataflow.communication.consumer.FileConsumer.FileConsumer (config={})
    Bases: pyDKB.dataflow.communication.consumer.Consumer
```

Data consumer implementation for HDFS data source.

```
current_file = None
get_source()
    Get nearest non-empty source (current or next).

get_source_info()
    Return current source info.

init_sources()
    Initialize sources iterator if not initialized yet.

next_source()
    Reset $current_file to the next non-empty file.

    Return value: File descriptor of the new $current_file None (no files left)

reconfigure(config={})
    (Re)initialize consumer with Stage configuration.

source_is_empty()
    Check if current source is empty.
```

pyDKB.dataflow.communication.consumer.HDFSConsumer module

Return value: True (empty) False (not empty) None (no source)

```
pyDKB.dataflow.communication.consumer.HDFSConsumer
```

```
 \textbf{class} \  \, \texttt{pyDKB.dataflow.communication.consumer.HDFSConsumer.HDFSConsumer} \, (\textit{config=\{\}}) \\  \, \textbf{Bases:} \, \, \textit{pyDKB.dataflow.communication.consumer.FileConsumer.FileConsumer} \, (\textbf{config=\{\}}) \\
```

Data consumer implementation for HDFS data source.

```
reconfigure (config={})
```

Configure HDFS Consumer according to the config parameters.

pyDKB.dataflow.communication.consumer.StreamConsumer module

pyDKB.dataflow.communication.consumer.StreamConsumer

Data consumer implementation for a single stream.

```
TODO: think about multiple streams (like a number of named pipes, etc). Prehaps, even merge this class with FileConsumer.
```

```
class pyDKB.dataflow.communication.consumer.StreamConsumer.StreamConsumer(config={})
    Bases: pyDKB.dataflow.communication.consumer.Consumer
```

Data consumer implementation for Stream data source.

```
fd = None
get_source()
    Get Stream file descriptor.

get_source_info()
    Return current source info.

next_source()
    Return None.
    As currenty we believe that there is only one input stream
reconfigure(config={})
    (Re)configure Stream consumer.
```

pyDKB.dataflow.communication.producer package

```
Producer submodule init file.
```

```
class pyDKB.dataflow.communication.producer.ProducerBuilder(config={})
Bases: object

Constructor for Producer instance.

build(config={})
Return constructed producer.

message_type = None
producerClass = None
setDest(dest)
Set data destination for the producer.

setSourceInfoMethod(src_info)
Set method to get current source info.

setType(Type)
Set message type for the producer.

src info = None
```

Submodules

pyDKB.dataflow.communication.producer.FileProducer module

pyDKB.dataflow.communication.producer.FileProducer

Data producer implementation for common (static) files.

TODO: think about:

• pipes (better, from the point of StreamProducer)

```
• multiple parallel dests
class pyDKB.dataflow.communication.producer.FileProducer.FileProducer(config={/})
     Bases: pyDKB.dataflow.communication.producer.Producer.Producer
     Data producer implementation for local file data dest.
     close()
          Close opened files and remove temporary one.
     close_file()
          Close current file.
     config_dir(config={})
          Configure output directory.
     current_file = None
     default_dir()
          Get default directory name.
     dirname (dirname=None)
          Set/get preferable directory name.
     ensure dir()
          Ensure that current directory for output files exists.
     file info()
          Return output file metadata (name, directory, ...).
     get_dest()
          Get destination file descriptor.
     get dest info()
          Get current destination info.
     get_dir()
          Get current directory for output files.
     get_filename()
          Return filename, corresponding the source, or timestamp-based.
     get source info()
          Set current data source, if any.
     reconfigure (config={})
          (Re)configure producer according to the config hash.
     reset_file()
          Resets current file according to the current source info.
          Metadata include:
                • fd – open file descriptor
                • name - file name
                • dir – directory name
                • local_path – local path to the file
     set_default_dir()
          Set default directory name.
```

```
subdir(base dir, sub dir=")
          Construct full path for $subdir of $base_dir.
pyDKB.dataflow.communication.producer.HDFSProducer module
pyDKB.dataflow.communication.producer.HDFSProducer
Data producer implementation for common (static) files in HDFS.
TODO: think about:
        • pipes (better, from the point of StreamProducer)
        • multiple parallel dests
class pyDKB.dataflow.communication.producer.HDFSProducer(config={/})
     Bases: pyDKB.dataflow.communication.producer.FileProducer.FileProducer
     Data producer implementation for HDFS data dest.
     close file()
          Close current file and move it to HDFS.
     config_dir(config={})
          Configure output directory.
     ensure_dir()
         Ensure that current directory for output files exists.
          Return output file metadata (name, directory, ...).
     set_default_dir()
          Set default directory name.
     subdir (base_dir, sub_dir=")
          Construct full path for $sub_dir of $base_dir.
pyDKB.dataflow.communication.producer.Producer module
pyDKB.dataflow.communication.producer.Producer
class pyDKB.dataflow.communication.producer.Producer.Producer(config={/})
     Bases: object
     Data producer implementation.
     close()
```

1.2. Subpackages 11

Close opened data stream and data dest.

Write EOP marker to the current dest.

Flush buffered messages to the current dest.

config = None

Drop buffered messages.

drop()

eop()

flush()

```
get dest()
          Return current destination.
     get_dest_info()
          Return current dest info.
     get stream(actualize=True)
          Get output stream linked to the current dest.
          If $actualize parameter set to True, will try to reset current stream destination; else will use last known
          destination or None.
     init stream()
          Init output stream (without real destination).
     log(message, level=3)
          Output log message with given log level.
     message_class()
          Return message class.
     message_type = None
     reconfigure (config={})
          (Re)initialize producer with stage config arguments.
     reset_stream()
          Reset input stream to the current dest.
     set_message_type (Type)
          Set input message type.
     write (msg)
          Put new message to the current dest (buffer).
exception pyDKB.dataflow.communication.producer.Producer.ProducerException
     Bases: pyDKB.dataflow.exceptions.DataflowException
     Dataflow Producer exception.
pyDKB.dataflow.communication.producer.StreamProducer module
pyDKB.dataflow.communication.producer.StreamProducer
Data producer implementation for a single stream.
TODO: think about multiple streams (like a number of named pipes, etc). Prehaps, even merge this class with
     FileProducer.
class pyDKB.dataflow.communication.producer.StreamProducer.StreamProducer(config={/})
     Bases: pyDKB.dataflow.communication.producer.Producer.Producer
     Data producer implementation for Stream data dest.
     fd = None
     get dest()
          Get Stream file descriptor.
     get_dest_info()
          Return current dest info.
```

```
pyDKB.dataflow.communication.stream package
pyDKB.dataflow.communication.stream
class pyDKB.dataflow.communication.stream.StreamBuilder(fd, config={})
     Bases: object
     Constructor for Stream object.
     build(config={})
          Create instance of Stream.
     message_type = None
     setStream(stream)
          Set stream type: 'input' or 'output'.
     setType (Type)
          Set message type for the Stream.
     streamClass = None
class pyDKB.dataflow.communication.stream.Stream(fd=None, config={})
     Bases: object
     Abstract class for input/output streams.
     close()
          Close open file descriptors etc.
     configure (config)
          Stream configuration.
     get_fd()
          Return open file descriptor or raise exception.
     log(message, level=3)
          Output log message with given log level.
     message_type()
          Get type of the messages in the stream.
     reset (fd, close=True)
          Reset file descriptor in operation.
              Parameters fd – open file descriptor TODO: IOBase objects
     set_message_type (msg_type)
          Set type of the messages in the stream.
class pyDKB.dataflow.communication.stream.InputStream(fd=None, config={})
     Bases: pyDKB.dataflow.communication.stream.Stream.Stream
     Implementation of the input stream.
     get_message()
          Get next message from the input stream.
          Return values: Message object False (failed to parse message) None (no messages left)
```

reconfigure (config={})

(Re)configure Stream producer.

```
next()
          Get next message from the input stream.
     parse_message (message)
          Verify and parse input message.
          Retrun value: Message object False (failed to parse)
     reset (fd, close=True)
          Reset current stream with new file descriptor.
          Overrides parent method to reset __iterator property.
class pyDKB.dataflow.communication.stream.OutputStream(fd=None, config=\{\})
     Bases: pyDKB.dataflow.communication.stream.Stream.Stream
     Implementation of the output stream.
     configure (config={})
          Configure instance.
     drop()
          Drop buffer without sending messages anywhere.
          Signalize Supervisor about end of process.
     flush()
          Flush buffer to the output stream.
     msg_buffer = []
     write (message)
          Add message to the buffer.
Submodules
pyDKB.dataflow.communication.stream.InputStream module
py DKB. data flow. communication. stream. Input Stream\\
class pyDKB.dataflow.communication.stream.InputStream.InputStream(fd=None,
                                                                                      config=\{\}
     Bases: pyDKB.dataflow.communication.stream.Stream.Stream
     Implementation of the input stream.
     get message()
          Get next message from the input stream.
          Return values: Message object False (failed to parse message) None (no messages left)
     next()
          Get next message from the input stream.
     parse_message(message)
          Verify and parse input message.
          Retrun value: Message object False (failed to parse)
     reset (fd, close=True)
          Reset current stream with new file descriptor.
```

Overrides parent method to reset __iterator property.

pyDKB.dataflow.communication.stream.OutputStream module

```
pyDKB.dataflow.communication.stream.OutputStream
class pyDKB.dataflow.communication.stream.OutputStream.OutputStream(fd=None,
                                                                                        con-
                                                                                        fig=\{\}
     Bases: pyDKB.dataflow.communication.stream.Stream.Stream
     Implementation of the output stream.
     configure (config={})
          Configure instance.
     drop()
          Drop buffer without sending messages anywhere.
     eop()
          Signalize Supervisor about end of process.
     flush()
          Flush buffer to the output stream.
     msg_buffer = []
     write (message)
          Add message to the buffer.
pyDKB.dataflow.communication.stream.Stream module
pyDKB.dataflow.commuication.stream.Stream
class pyDKB.dataflow.communication.stream.Stream (fd=None, config={})
     Bases: object
     Abstract class for input/output streams.
     close()
          Close open file descriptors etc.
     configure (config)
          Stream configuration.
     get_fd()
          Return open file descriptor or raise exception.
     log(message, level=3)
          Output log message with given log level.
     message_type()
          Get type of the messages in the stream.
     reset (fd, close=True)
          Reset file descriptor in operation.
              Parameters fd – open file descriptor TODO: IOBase objects
     set_message_type (msg_type)
          Set type of the messages in the stream.
```

pyDKB.dataflow.communication.stream.exceptions module

```
pyDKB.dataflow.communication.stream.exceptions
```

exception pyDKB.dataflow.communication.stream.exceptions.StreamException Bases: pyDKB.dataflow.exceptions.DataflowException

Exception for Stream operations.

Submodules

pyDKB.dataflow.communication.messages module

```
pyDKB.dataflow.communication.messages
```

Bases: exceptions.NotImplementedError

Exception to be thrown when message type is not encodable.

```
Definition of abstract message class and specific message classes
class pyDKB.dataflow.communication.messages.AbstractMessage (message=None)
     Bases: object
     Abstract message
     content()
         Return message content.
     decode (code)
         Decode original from CODE to TYPE-specific format.
         Raises ValueError
     decoded = None
     encode (code)
         Encode original message from TYPE-specific format to CODE.
         Raises ValueError
     encoded = None
     classmethod extension()
         Return file extension corresponding this message type.
     getOriginal()
         Return original message.
     msg_type = None
     native_types = []
     classmethod typeName()
         Return message type name as string.
exception pyDKB.dataflow.communication.messages.DecodeUnknownType (code, cls)
     Bases: exceptions.NotImplementedError
     Exception to be thrown when message type is not decodable.
exception pyDKB.dataflow.communication.messages.EncodeUnknownType (code, cls)
```

```
class pyDKB.dataflow.communication.messages.JSONMessage (message=None)
     Bases: pyDKB.dataflow.communication.messages.AbstractMessage
     Message in JSON format.
     decode(code=1)
         Decode original data as JSON.
     encode(code=1)
         Encode JSON as CODE.
     msg\_type = 2
     native_types = [<type 'dict'>]
pyDKB.dataflow.communication.messages.Message(msg_type)
     Return class XXXMessage, where XXX is the passed type.
class pyDKB.dataflow.communication.messages.TTLMessage(message=None)
     Bases: pyDKB.dataflow.communication.messages.AbstractMessage
     Messages in TTL format
     Single message = single TTL statement
     decode(code=1)
         Decode original data as TTL.
         Currently takes text as it is. TODO: check some formal matter to confirm the string is TTL.
     encode (code=1)
         Encode TTL as CODE.
     msg\_type = 3
     native_types = [<type 'str'>, <type 'unicode'>]
pyDKB.dataflow.stage package
Stage submodule init file.
class pyDKB.dataflow.stage.ProcessorStage (description='DKB Dataflow data processing
                                                    stage.')
     Bases: pyDKB.dataflow.stage.AbstractStage.AbstractStage
     Abstract class to implement Processor stages
     Processor stage – is a stage for data processing/transformation.
     Class/instance variable description:
        • communication.consumer.Consumer instance __input
        • Generator object for output file descriptor OR file descriptor (for (s)tream mode)
              __output
        • List of objects to be "stopped" __stoppable
     clear buffer()
         Drop buffered output messages.
     configure (args=None)
         Configure stage according to the config parameters.
         If $args specified, arguments will be parsed anew.
```

```
defaultArguments()
     Default parser configuration.
flush buffer()
     Flush message buffer to the output.
forward()
     Send EOPMarker to the output stream.
get_out_stream()
     Get current output stream.
get_source_info()
     Get information about current source.
input()
     Generator for input messages.
     Returns iterable object. Every iteration returns single input message to be processed.
input_message_class()
     Get input message class.
output (message)
     Put the (list of) message(s) to the output buffer.
output_message_class()
     Get output message class.
static process(stage, input_message)
     Transform input_message -> output_message.
     To be implemented individually for every stage. Takes the stage as first argument to allow calling output()
         from inside the function.
     Return value: True – processing successfully finished False – processing failed (skip the input message)
run()
     Run process() for every input() message.
set_input_message_type(Type=None)
     Set input message type.
set_output_message_type (Type=None)
     Set output message class.
stop()
     Finalize all the processes and prepare to exit.
```

Submodules

pyDKB.dataflow.stage.AbstractStage module

Definition of an abstract class for Dataflow Stages.

```
class pyDKB.dataflow.stage.AbstractStage.AbstractStage (description='DKB Dataflow
                                                              stage')
    Bases: object
```

Class/instance variable description: * Argument parser (argparse.ArgumentParser)

```
    Parsed arguments (argparse.Namespace) ARGS

   • Stage config parser (ConfigParser.SafeConfigParser) __config
   • Stage custom config (defaultdict(defaultdict(str))) CONFIG
add_argument (*args, **kwargs)
     Add specific (not common) arguments.
args_error (message)
     Output USAGE, error message and exit with code 2.
config_error (message='Failed to read config file:')
     Output error message and exit with code 3.
defaultArguments()
     Config argument parser with parameters common for all stages.
log(message, level=3)
     Output log message with given log level.
output_error (message=None, exc_info=None)
     Output traceback of the passed (or last) error with message.
parse_args (args)
     Parse arguments and set dependant arguments if needed.
     Exits in case of error with code: 2 – failed to parse arguments 3 – failed to read config file
print_usage (fd=<open file '<stderr>', mode 'w'>)
     Print usage message.
read config()
     Reads stage custom config file.
         Returns (TruelFalse)
run()
     Run the stage.
set_error (err_type, err_val, err_trace)
     Set object err variable from the last error info.
stop()
```

pyDKB.dataflow.stage.ProcessorStage module

Definition of an abstract class for Dataflow Data Processing Stages.

Stop running processes and output error information.

USAGE: ProcessorStage [<options>] [<input files>]

OPTIONS:

__parser

-s,source	{flslh} - where to get data from: local (f)iles, (s)tdin, (h)dfs
-i,input-dir	DIR - base directory for relative input file names (for local and HDFS sources). If <input files=""/> not specified, all files from the directory will be taken as the input.
-ddest	{f s h} - where to send data to: local (f)iles, (s)tdout, (h)dfs

```
-o, --output-dir
                                     DIR - base directory for output files (for local and HDFS
                                     sources)
                 --hdfs
                                       • equivalent to "-source h -dest h"
                 -m, --mode
                                     MODE - MODE: (f)ile = -source f
                                         -dest f (can be
                                            rewritten with 's' or 'h')
                                     (s)tream = -source s (can be
                                            rewritten with 'h')
                                         -dest s
                                     (m)apreduce = -source s (can be
                                            rewritten with 'h')
                                         -dest s
class pyDKB.dataflow.stage.ProcessorStage.ProcessorStage(description='DKB)
                                                                              Dataflow data processing
                                                                              stage.')
     Bases: pyDKB.dataflow.stage.AbstractStage.AbstractStage
     Abstract class to implement Processor stages
     Processor stage – is a stage for data processing/transfornation.
     Class/instance variable description:
        • communication.consumer.Consumer instance __input
        • Generator object for output file descriptor OR file descriptor (for (s)tream mode)
               __output
        • List of objects to be "stopped" __stoppable
     clear buffer()
          Drop buffered output messages.
     configure (args=None)
          Configure stage according to the config parameters.
          If $args specified, arguments will be parsed anew.
     defaultArguments()
          Default parser configuration.
     flush_buffer()
          Flush message buffer to the output.
     forward()
          Send EOPMarker to the output stream.
     get_out_stream()
          Get current output stream.
     get source info()
          Get information about current source.
     input()
          Generator for input messages.
```

```
Returns iterable object. Every iteration returns single input message to be processed.
     input_message_class()
          Get input message class.
     output (message)
          Put the (list of) message(s) to the output buffer.
     output_message_class()
          Get output message class.
     static process (stage, input_message)
          Transform input_message -> output_message.
          To be implemented individually for every stage. Takes the stage as first argument to allow calling output()
              from inside the function.
          Return value: True – processing successfully finished False – processing failed (skip the input message)
     run()
          Run process() for every input() message.
     set_input_message_type (Type=None)
          Set input message type.
     set_output_message_type (Type=None)
          Set output message class.
     stop()
          Finalize all the processes and prepare to exit.
Submodules
pyDKB.dataflow.cds module
Extended CDSInvenioConnector allowing us to login via Kerberos
class pyDKB.dataflow.cds.CDSInvenioConnector(*args)
     Bases: invenio_client.contrib.cds.CDSInvenioConnector
     CDSInvenioConnector which closes the browser in most cases.
     delete (restore_handlers=True)
     handlers = False
     kill (signum, frame)
          Run del and propagate signal.
     orig_handlers = {}
class pyDKB.dataflow.cds.KerberizedCDSInvenioConnector(login='user',
                                                                                             pass-
```

1.2. Subpackages 21

Represents same CDSInvenioConnector, but this one is aware about SPNEGO: Simple and Protected GSSAPI

Bases: pyDKB.dataflow.cds.CDSInvenioConnector

Negotiation Mechanism

word='password')

pyDKB.dataflow.dkbID module

Utils to generate unique yet meaningful identifier for DKB objects.

```
pyDKB.dataflow.dkbID.dkbID (json_data, data_type)
Return unique identifier for object of TYPE based on DATA.
```

pyDKB.dataflow.exceptions module

Definition of DKB Dataflow exceptions

```
exception pyDKB.dataflow.exceptions.DataflowException
Bases: exceptions.Exception
Base Exception for Dataflow modules.
```

pyDKB.dataflow.types module

Type definitions for library objects.

CHAPTER

TWO

Indices and tables

- genindex
- modindex
- search

PYTHON MODULE INDEX

```
р
                                         pyDKB.dataflow.dkbID, 22
                                         pyDKB.dataflow.exceptions, 22
pyDKB, 1
                                         pyDKB.dataflow.stage, 17
pyDKB.common, 3
                                         pyDKB.dataflow.stage.AbstractStage, 18
pyDKB.common.custom_readline, 4
                                         pyDKB.dataflow.stage.ProcessorStage, 19
pyDKB.common.exceptions, 4
                                         pyDKB.dataflow.types, 22
pyDKB.common.hdfs,4
pyDKB.common.json_utils,5
pyDKB.common.Type, 3
pyDKB.dataflow, 6
pyDKB.dataflow.cds, 21
pyDKB.dataflow.communication, 6
pyDKB.dataflow.communication.consumer,
pyDKB.dataflow.communication.consumer.Consumer,
pyDKB.dataflow.communication.consumer.FileConsumer,
pyDKB.dataflow.communication.consumer.HDFSConsumer,
pyDKB.dataflow.communication.consumer.StreamConsumer,
pyDKB.dataflow.communication.messages,
pyDKB.dataflow.communication.producer,
pyDKB.dataflow.communication.producer.FileProducer,
pyDKB.dataflow.communication.producer.HDFSProducer,
pyDKB.dataflow.communication.producer.Producer,
pyDKB.dataflow.communication.producer.StreamProducer,
pyDKB.dataflow.communication.stream, 13
pyDKB.dataflow.communication.stream.exceptions,
pyDKB.dataflow.communication.stream.InputStream,
pyDKB.dataflow.communication.stream.OutputStream,
pyDKB.dataflow.communication.stream.Stream,
       15
```

26 Python Module Index

INDEX

Α					config (py	DKB.dataflo	ow.communi	cation.consume	er.Consumer.Con	nsumer
Abstract	Message	(class	in	ру-		attribute), 7				
	-	communication.	n.messages),					cation.producer	.Producer.Produ	icer
	16					attribute), 1				
Abstract	-	(class	in	py-				nmunication.pro	oducer.FileProdu	acer.FilePi
		stage.AbstractS	-			method), 10			1 HDECD	1 1115
add() (py	DKB.common.	.Type.Type meth	nod), 3		_			imunication.pro	oducer.HDFSPro	oducer.HL
add_argu	ment() (pyDKF method), 19	B.dataflow.stage	.AbstractStag	e.Abs	tractStage config_err	method), 11 or() (pyDK)	B.dataflow.st	age.AbstractSta	nge.AbstractStag	ge
args erro		taflow.stage.Abs	stractStage.Al	bstrac	tStage	method), 19)			
8-2-	method), 19				configure(() (pyDKB.d	ataflow.com	munication.stre	am.OutputStrea	m
_	,,					method), 14				
В					_			munication.stre	am.OutputStrea	m.Output
basenam	e() (in module p	yDKB.commo	n.hdfs), 5			method), 15	~		~	
build() (p	yDKB.dataflov	v.communicatio	n.consumer.C	Consur	configure(nerBuilder	() (pyDKB.d method), 13	ataflow.com	munication.stre	am.Stream	
huild() (r	vDKR dataflov	v.communicatio	n producer Pr	oduce	configure(() (pyDKB.d	ataflow.com	munication.stre	am.Stream.Strea	am
ound() (I	method), 9	v.communicatio	ii.producei.i i	oduce	a Dunder	method), 15				
build() (r	vDKB.dataflov	v.communicatio	n.stream.Stre	amBu	ilder	() (py)		w.stage.Process	orStage	
(I	method), 13					memoa), 17				
					_		_	e.ProcessorStag	e.ProcessorStag	ge
С						method), 20				
CDSInve	enioConnector (class in pyDKB	.dataflow.cds), 21	Consumer		(class	in	py-	
		le pyDKB.comr		,,		DKB.datano	ow.communi	cation.consume	r.Consumer),	
clear_bu		KB.dataflow.stag		tage	Consumer	/ :Duildar	(class	in	DV.	
	method), 17	_				DKB datafle	,	cation.consume	py-	
clear_bu	ffer() (pyDKB.d	lataflow.stage.Pr	rocessorStage	.Proc	essorStage	6	Jw.Commun	cation.consume	1),	
	method), 20				consumer	o Class (nvDK	B dataflow	communication	consumer Cons	umerBuild
close() (p	yDKB.dataflov	v.communicatio	n.consumer.C	Consur	ner.Consur	ner attribute), 6	iD.datano w.		consumor.coms	uniciBun
	memou), /				C	. [7			
close() (p	yDKB.dataflov	v.communicatio	n.producer.Fi	lePro	ducer FileP content() (roducer pyDKB.dat	aflow.comm	unication.messa	iges.AbstractMe	essage
close() (p	yDKB.dataflov	v.communicatio	n.producer.Pr	oduce	er.Producer current_fi	le (pyDKB.c	dataflow.com	munication.con	sumer.FileCons	sumer.File
	1110 (1110 (1), 11					attribute), 8				
close()	method), 13	ow.communicati	on.siream.sir	eam				munication.pro	ducer.FileProdu	ıcer.FilePr
close() (r	wDKR dataflow	v.communicatio	n straam Stra	am St	raam	attribute), 1	0			
ciose() (I	method), 15	v.communicatio	n.sucam.suc	am.st	custom_re	eadline()	(in	module	ру-	
close file	e() (pvDKB data	aflow.communic	cation produc	er File	Producer F	DKB.comm ileProducer	ion.custom_1	readline), 4		
2100 0 _III	method), 10	and mooning and	omproduc		D					
close_file	//	aflow.communio	cation.produc	er.HD	FSProduce	r.HDFSProc	lucer			
_	method), 11		1		Dataflow	exception, 2	2			

```
decode() (pyDKB.dataflow.communication.messages.AbstractMessage (pyDKB.dataflow.communication.messages.AbstractMessage
                                                            method), 16
                                                                                                                                                                                                                                                                                                                                                                                                                                   class method), 16
decode() (pyDKB.dataflow.communication.messages.JSONMessage
                                                            method), 17
 decode() (pyDKB.dataflow.communication.messages.TTLMescage) KB.dataflow.communication.consumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.S
                                                           method), 17
                                                                                                                                                                                                                                                                                                                                                                                                                                    attribute), 9
attribute), 16
                                                                                                                                                                                                                                                                                                                                                                                                                                    attribute), 12
DecodeUnknownType, 16
                                                                                                                                                                                                                                                                                                                                                                       File() (in module pyDKB.common.hdfs), 4
default\_dir()\ (pyDKB.data flow.communication.producer. File \cite{Communication} File \cite{C
                                                            method), 10
                                                                                                                                                                                                                                                                                                                                                                                                                                   method), 10
defaultArguments()
                                                                                                                                                                                                                                                                                                                                (py-
                                                                                                                                                                                                                                                                                                                                                                      file info() (pyDKB.dataflow.communication.producer.HDFSProducer.HDF
                                                            DKB.dataflow.stage.AbstractStage.AbstractStage
                                                                                                                                                                                                                                                                                                                                                                                                                                   method), 11
                                                            method), 19
                                                                                                                                                                                                                                                                                                                                                                      FileConsumer
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              (class
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                in
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            ру-
defaultArguments()
                                                                                                                                                                                                                                                                                                                                (py-
                                                                                                                                                                                                                                                                                                                                                                                                                                   DKB.dataflow.communication.consumer.FileConsumer),
                                                           DKB.dataflow.stage.ProcessorStage method),
                                                                                                                                                                                                                                                                                                                                                                                                                                    8
                                                            18
                                                                                                                                                                                                                                                                                                                                                                      FileProducer
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        (class
defaultArguments()
                                                                                                                                                                                                                                                                                                                                (py-
                                                                                                                                                                                                                                                                                                                                                                                                                                   DKB.dataflow.communication.producer.FileProducer),
                                                          DKB.dataflow.stage.ProcessorStage.ProcessorStage
                                                           method), 20
                                                                                                                                                                                                                                                                                                                                                                       flush() (pyDKB.dataflow.communication.producer.Producer.Producer
delete()
                                                                           (pyDKB.dataflow.cds.CDSInvenioConnector
                                                                                                                                                                                                                                                                                                                                                                                                                                    method), 11
                                                           method), 21
                                                                                                                                                                                                                                                                                                                                                                       flush() (pyDKB.dataflow.communication.stream.OutputStream
dirname() (in module pyDKB.common.hdfs), 5
                                                                                                                                                                                                                                                                                                                                                                                                                                   method), 14
dirname() (pyDKB.dataflow.communication.producer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProducer.FileProd
                                                            method), 10
                                                                                                                                                                                                                                                                                                                                                                                                                                   method), 15
dkbID() (in module pyDKB.dataflow.dkbID), 22
                                                                                                                                                                                                                                                                                                                                                                       flush_buffer()
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     (pyDKB.dataflow.stage.ProcessorStage
drop() (pyDKB.dataflow.communication.producer.Producer Producer method), 18
                                                            method), 11
                                                                                                                                                                                                                                                                                                                                                                      flush buffer() (pyDKB.dataflow.stage.ProcessorStage.ProcessorStage
drop() \ (pyDKB. data flow. communication. stream. Output Stream
                                                                                                                                                                                                                                                                                                                                                                                                                                   method), 20
                                                            method), 14
                                                                                                                                                                                                                                                                                                                                                                       forward()
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     (pyDKB.dataflow.stage.ProcessorStage
drop() (pyDKB.dataflow.communication.stream.OutputStream.OutputStream.), 18
                                                            method), 15
                                                                                                                                                                                                                                                                                                                                                                      forward() (pyDKB.dataflow.stage.ProcessorStage.ProcessorStage
                                                                                                                                                                                                                                                                                                                                                                                                                                   method), 20
 Ε
encode() (pyDKB.dataflow.communication.messages.Abstraction.message
                                                            method), 16
                                                                                                                                                                                                                                                                                                                                                                       get\_dest() \ (pyDKB. data flow. communication. producer. File Pr
encode() (pyDKB.dataflow.communication.messages.JSONMessage method), 10
                                                            method), 17
                                                                                                                                                                                                                                                                                                                                                                       get dest() (pyDKB.dataflow.communication.producer.Producer.Producer
encode() (pyDKB.dataflow.communication.messages.TTLMessage
                                                                                                                                                                                                                                                                                                                                                                                                                                 method), 11
                                                            method), 17
                                                                                                                                                                                                                                                                                                                                                                      get_dest() (pyDKB.dataflow.communication.producer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.StreamProducer.S
encoded (pyDKB.dataflow.communication.messages.AbstractMessagemethod), 12
                                                            attribute), 16
                                                                                                                                                                                                                                                                                                                                                                       get_dest_info() (pyDKB.dataflow.communication.producer.FileProducer.Fil
EncodeUnknownType, 16
                                                                                                                                                                                                                                                                                                                                                                                                                                   method), 10
ensure\_dir() \ (pyDKB.dataflow.communication.producer.File \cite{Producer.File} \cite{Produ
                                                            method), 10
                                                                                                                                                                                                                                                                                                                                                                                                                                   method), 12
ensure\_dir() \ (pyDKB.dataflow.communication.producer. HDFSPFPQucerFHDFSPFRB1988 faflow.communication.producer. Stream Producer. HDFSPFRB1988 faflow.communication.producer. HDFSPFRB1988 faflow.communication.produ
                                                           method), 11
                                                                                                                                                                                                                                                                                                                                                                                                                                    method), 12
eop()\ (pyDKB. data flow. communication. producer. Pro
                                                            method), 11
                                                                                                                                                                                                                                                                                                                                                                                                                                   method), 10
eop() \ (pyDKB. data flow. communication. stream. Output Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream. Stream \ fd() \ (pyDKB. data flow. communication. stream \ fd() \ (
                                                            method), 14
                                                                                                                                                                                                                                                                                                                                                                                                                                    method), 13
eop()\ (pyDKB. data flow. communication. stream. Output Stream \underline{Output}\ Stream. St
                                                            method), 15
                                                                                                                                                                                                                                                                                                                                                                                                                                    method), 15
```

```
get filename() (pyDKB.dataflow.communication.producer.FibiProducer.FibiProducer.Froducer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Prod
                                    method), 10
                                                                                                                                                                                                                                                                   method), 12
get message() (pyDKB.dataflow.communication.consumer.Gipnst@metpQDKSB.ndataflow.stage.ProcessorStage method),
                                    method), 7
get_message() (pyDKB.dataflow.communication.stream.InpintStrt()atpyDKB.dataflow.stage.ProcessorStage.ProcessorStage
                                    method), 13
                                                                                                                                                                                                                                                                   method), 20
get message() (pyDKB.dataflow.communication.stream.InpittStrteamessagetSchassn)
                                                                                                                                                                                                                                                                                                                                                                                                                                   (py-
                                                                                                                                                                                                                                                                   DKB.dataflow.stage.ProcessorStage method),
                                     method), 14
get_out_stream() (pyDKB.dataflow.stage.ProcessorStage
                                     method), 18
                                                                                                                                                                                                                             input_message_class()
get_out_stream() (pyDKB.dataflow.stage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorStage.ProcessorSta
                                     method), 20
                                                                                                                                                                                                                                                                   method), 21
get_source() (pyDKB.dataflow.communication.consumer.Fille@outStreamFileConsumerlass
                                                                                                                                                                                                                                                                                                                                                                                     in
                                                                                                                                                                                                                                                                                                                                                                                                                                      ру-
                                    method), 8
                                                                                                                                                                                                                                                                   DKB.dataflow.communication.stream), 13
get_source() (pyDKB.dataflow.communication.consumer.StrleamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsumer.StreamConsume
                                                                                                                                                                                                                                                                                                                                                                                     in
                                                                                                                                                                                                                                                                                                                                                                                                                                      ру-
                                     method), 9
                                                                                                                                                                                                                                                                   DKB.dataflow.communication.stream.InputStream),
get_source_info() (pyDKB.dataflow.communication.consumer.Consumer
get_source_info() (pyDKB.dataflow.communication.consumer.FileConsumer.FileConsumer
                                                                                                                                                                                                                             ioin() (in module pvDKB.common.hdfs), 5
                                     method). 8
get_source_info() (pyDKB.dataflow.communication.consumts@neamscapsumer.Streamscapsumer.
                                                                                                                                                                                                                                                                                                                                                                                                                                     ру-
                                    method), 9
                                                                                                                                                                                                                                                                  DKB.dataflow.communication.messages),
get\_source\_info() \, (pyDKB. data flow. communication. producer. File Producer. 
                                     method), 10
get source info()(pyDKB.dataflow.stage.ProcessorStage
                                     method), 18
                                                                                                                                                                                                                              KerberizedCDSInvenioConnector
                                                                                                                                                                                                                                                                                                                                                                          (class
                                                                                                                                                                                                                                                                                                                                                                                                               in
                                                                                                                                                                                                                                                                                                                                                                                                                                      ру-
get_source_info() (pyDKB.dataflow.stage.ProcessorStage.ProcessorStage.B.dataflow.cds), 21
                                     method), 20
                                                                                                                                                                                                                                                                            (pyDKB.dataflow.cds.CDSInvenioConnector
                                                                                                                                                                                                                             kill()
get_stream() (pyDKB.dataflow.communication.consumer.Consumer.Consumer.21
get_stream() (pyDKB.dataflow.communication.producer.Producer
                                     method), 12
                                                                                                                                                                                                                             listdir() (in module pyDKB.common.hdfs), 5
getfile() (in module pyDKB.common.hdfs), 5
                                                                                                                                                                                                                             log() (pyDKB.dataflow.communication.consumer.Consumer.Consumer
getOriginal() (pyDKB.dataflow.communication.messages.AbstractMessagesod), 7
                                     method), 16
                                                                                                                                                                                                                             log() (pyDKB.dataflow.communication.producer.Producer.Producer
                                                                                                                                                                                                                                                                   method), 12
Η
                                                                                                                                                                                                                             log()
                                                                                                                                                                                                                                                              (pyDKB.dataflow.communication.stream.Stream
handlers (pyDKB.dataflow.cds.CDSInvenioConnector at-
                                                                                                                                                                                                                                                                   method), 13
                                     tribute), 21
                                                                                                                                                                                                                             log() (pyDKB.dataflow.communication.stream.Stream.Stream
hasMember() (pyDKB.common.Type.Type method), 3
                                                                                                                                                                                                                                                                   method), 15
HDFSConsumer
                                                                                                  (class
                                                                                                                                                            in
                                                                                                                                                                                                                            log() (pyDKB.dataflow.stage.AbstractStage.AbstractStage
                                    DKB.dataflow.communication.consumer.HDFSConsumer), method), 19
                                                                                                                                                                                                                              Μ
HDFSException, 4
HDFSProducer
                                                                                               (class
                                                                                                                                                                                                        ру-
                                                                                                                                                                                                                            makedirs() (in module pyDKB.common.hdfs), 5
                                     DKB.dataflow.communication.producer.HDFSProducer() (pyDKB.common.Type.Type method), 3
                                     11
                                                                                                                                                                                                                             memberName() (pyDKB.common.Type.Type method), 3
                                                                                                                                                                                                                             Message() (in module pyDKB.dataflow.communication),
init_sources() (pyDKB.dataflow.communication.consumer.FileConsumer
                                                                                                                                                                                                                                                                                                                                                                module
                                                                                                                                                                                                                                                                                                                                                                                                                                      ру-
                                     method), 8
                                                                                                                                                                                                                                                                   DKB.dataflow.communication.messages),
init stream() (pyDKB.dataflow.communication.consumer.Consumer.Consumer
                                     method), 7
                                                                                                                                                                                                                             message_class() (pyDKB.dataflow.communication.consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Consumer.Co
                                                                                                                                                                                                                                                                   method), 7
```

```
message class() (pyDKB.dataflow.communication.producer@rtpdutcerdssadecctass()
                                                                                                                                                                                                                                                                                                                                                      (py-
                               method), 12
                                                                                                                                                                                                                  DKB.dataflow.stage.ProcessorStage method),
message type (pyDKB.dataflow.communication.consumer.Consumer.Consumer
                               attribute), 7
                                                                                                                                                                                    output message class()
                                                                                                                                                                                                                                                                                                                                                      (py-
message type (pyDKB.dataflow.communication.producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Producer.Produ
                              attribute), 12
                                                                                                                                                                                                                  method), 21
message type (pyDKB.dataflow.communication.producer.PfOdtpoetStreikfer
                                                                                                                                                                                                                                                                (class
                                                                                                                                                                                                                                                                                                                 in
                                                                                                                                                                                                                                                                                                                                                        ру-
                               attribute), 9
                                                                                                                                                                                                                   DKB.dataflow.communication.stream), 14
message_type (pyDKB.dataflow.communication.stream.StreamButBleeam
                                                                                                                                                                                                                                                                (class
                                                                                                                                                                                                                                                                                                                 in
                                                                                                                                                                                                                                                                                                                                                        ру-
                               attribute), 13
                                                                                                                                                                                                                   DKB.dataflow.communication.stream.OutputStream),
message_type() (pyDKB.dataflow.communication.stream.Stream
                               method), 13
message_type() (pyDKB.dataflow.communication.stream.Stream.Stream
                               method), 15
                                                                                                                                                                                    parse_args() (pyDKB.dataflow.stage.AbstractStage.AbstractStage
movefile() (in module pyDKB.common.hdfs), 5
                                                                                                                                                                                                                   method), 19
msg\_buffer (pyDKB. dataflow. communication. stream. Output Stream message () (pyDKB. dataflow. communication. stream. Input Stream message () (pyDKB. dataflow. communication. stream message () (pyDKB. dataflow. communicat
                              attribute), 14
                                                                                                                                                                                                                   method), 14
msg buffer (pyDKB.dataflow.communication.stream.OutputStream.OutputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.InputStream.Inpu
                              attribute), 15
                                                                                                                                                                                                                   method), 14
msg_type (pyDKB.dataflow.communication.messages.AbstractStage.AbstractStage.AbstractStage
                              attribute), 16
                                                                                                                                                                                                                   method), 19
msg_type (pyDKB.dataflow.communication.messages.JSONMessage) (pyDKB.dataflow.stage.ProcessorStage static
                               attribute), 17
                                                                                                                                                                                                                   method), 18
msg_type (pyDKB.dataflow.communication.messages.TTLMassage) (pyDKB.dataflow.stage.ProcessorStage.ProcessorStage
                               attribute), 17
                                                                                                                                                                                                                   static method), 21
                                                                                                                                                                                    ProcessorStage (class in pyDKB.dataflow.stage), 17
Ν
                                                                                                                                                                                    ProcessorStage
                                                                                                                                                                                                                                                                 (class
                                                                                                                                                                                                                                                                                                                                                        ру-
native_types (pyDKB.dataflow.communication.messages.AbstractMesDKB.dataflow.stage.ProcessorStage), 20
                                                                                                                                                                                     Producer
                                                                                                                                                                                                                                                     (class
                               attribute), 16
                                                                                                                                                                                                                                                                                                                                                        ру-
native\_types~(pyDKB.dataflow.communication.messages.JSONMessageKB.dataflow.communication.producer.Producer), and the producer of the produce
                              attribute), 17
native_types (pyDKB.dataflow.communication.messages.TTPrMessageuilder
                                                                                                                                                                                                                                                                     (class
                                                                                                                                                                                                                                                                                                                    in
                                                                                                                                                                                                                                                                                                                                                        ру-
                                                                                                                                                                                                                   DKB.dataflow.communication.producer),
                              attribute), 17
nestedKeys() (in module pyDKB.common.json utils), 5
next() (pyDKB.dataflow.communication.consumer.Consumer@dusarGlass (pyDKB.dataflow.communication.producer.ProducerBuilder
                                                                                                                                                                                                                   attribute), 9
                               method), 7
next() \ (pyDKB. data flow. communication. stream. InputStream Producer Exception, \ 12 \\
                                                                                                                                                                                    putfile() (in module pyDKB.common.hdfs), 5
                               method), 13
next() (pyDKB.dataflow.communication.stream.InputStreamphDKtBt(randule), 1
                                                                                                                                                                                    pyDKB.common (module), 3
                               method), 14
next_source() (pyDKB.dataflow.communication.consumer.FireOKBunnerForeCustomereadline (module), 4
                                                                                                                                                                                    pyDKB.common.exceptions (module), 4
                               method), 8
next_source() (pyDKB.dataflow.communication.consumer.SpyeDKBcommuniShdefarh@odsulen)e4
                                                                                                                                                                                    pyDKB.common.json utils (module), 5
                               method), 9
                                                                                                                                                                                    pyDKB.common.Type (module), 3
0
                                                                                                                                                                                    pyDKB.dataflow (module), 6
orig_handlers (pyDKB.dataflow.cds.CDSInvenioConnector pyDKB.dataflow.cds (module), 21
                                                                                                                                                                                    pyDKB.dataflow.communication (module), 6
                              attribute), 21
                                                                                                                                                                                    pyDKB.dataflow.communication.consumer (module), 6
output() (pyDKB.dataflow.stage.ProcessorStage method),
                                                                                                                                                                                    pyDKB.dataflow.communication.consumer.Consumer
                                                                                                                                                                                                                   (module), 7
output() (pyDKB.dataflow.stage.ProcessorStage.ProcessorStage
                                                                                                                                                                                    pyDKB.dataflow.communication.consumer.FileConsumer
                              method), 21
                                                                                                                                                                                                                   (module), 8
output_error() (pyDKB.dataflow.stage.AbstractStage.AbstractStage
                              method), 19
```

	erreset() (pyDKB.dataflow.communication.stream.Stream.Stream
(module), 8	method), 15
	$\textbf{ex} \textbf{reset_file}() \ (pyDKB.dataflow.communication.producer.FileProducer.Fil$
(module), 8	method), 10
pyDKB.dataflow.communication.messages (module), 16	$reset_stream() \ (pyDKB. data flow. communication. consumer. Con$
pyDKB.dataflow.communication.producer (module), 9	method), 7
pyDKB.dataflow.communication.producer.FileProducer (module), 9	reset_stream() (pyDKB.dataflow.communication.producer.Producer.Producer.producer.Producer.pro
pyDKB.dataflow.communication.producer.HDFSProducer	
(module), 11	method), 19
pyDKB.dataflow.communication.producer.Producer	run() (pyDKB.dataflow.stage.ProcessorStage method), 18
(module), 11	run() (pyDKB. data flow. stage. Processor Stage. Processor Stage
py DKB. data flow. communication. producer. Stream Producer	method), 21
(module), 12	
pyDKB.dataflow.communication.stream (module), 13	S
pyDKB.dataflow.communication.stream.exceptions (module), 16	set_default_dir() (pyDKB.dataflow.communication.producer.FileProducer.F
pyDKB.dataflow.communication.stream.InputStream	method), 10
(module), 14	set_default_dir() (pyDKB.dataflow.communication.producer.HDFSProduce method), 11
pyDKB.dataflow.communication.stream.OutputStream (module), 15	set_error() (pyDKB.dataflow.stage.AbstractStage.AbstractStage method), 19
pyDKB.dataflow.communication.stream.Stream (mod-	set_input_message_type() (py-
ule), 15	
pyDKB.dataflow.dkbID (module), 22	DKB.dataflow.stage.ProcessorStage method),
pyDKB.dataflow.exceptions (module), 22	
pyDKB.dataflow.stage (module), 17	set_input_message_type() (py-
pyDKB.dataflow.stage (module), 17 pyDKB.dataflow.stage.AbstractStage (module), 18	DKB.dataflow.stage.ProcessorStage.ProcessorStage
	method), 21
pyDKB.dataflow.stage.ProcessorStage (module), 19	set_message_type() (py-
pyDKB.dataflow.types (module), 22	DKB.dataflow.communication.consumer.Consumer.Consumer
R	method), 7
	set_message_type() (py-
$read_config() \ (pyDKB.dataflow.stage.AbstractStage.Abst$	ctStage DKB.dataflow.communication.producer.Producer.Producer
method), 19	method), 12
reconfigure() (pyDKB.dataflow.communication.consumer.C	
method), 7	DKB.dataflow.communication.stream.Stream
reconfigure() (pyDKB.dataflow.communication.consumer.F	ileConsum eneFile Consumer
method), 8	set_message_type() (py-
reconfigure() (pyDKB.dataflow.communication.consumer.H	IDFSConsi DKeBHDFB @wwwmmnication.stream.Stream.Stream
method), 8	method), 15
reconfigure() (pyDKB.dataflow.communication.consumer.S	tretam Cityout umer Streea try Dech sumer (py-
method), 9	DKB.dataflow.stage.ProcessorStage method),
reconfigure() (pyDKB.dataflow.communication.producer.Fi	
method), 10	set_output_message_type() (py-
	roducer.ProDMcBrdataflow.stage.ProcessorStage.ProcessorStage
method), 12	method), 21
	rseathDest(dupyrLMtBathaPathdwccommunication.producer.ProducerBuilder
method), 12	method), 9
	rasetSource() (pyDKB.dataflow.communication.consumer.ConsumerBuilder
method), 14	method), 6
reset() (pyDKB.dataflow.communication.stream.InputStream	•
method), 14	DKB.dataflow.communication.producer.ProducerBuilder
reset() (pyDKB.dataflow.communication.stream.Stream	method), 9
method), 13	setStream() (pyDKB.dataflow.communication.stream.StreamBuilder
	method), 13

```
setType() (pyDKB.dataflow.communication.consumer.ConsumniteBuilble KB.dataflow.communication.stream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStream.OutputStr
                    method), 7
                                                                                                                                           method), 15
setType() (pyDKB.dataflow.communication.producer.ProducerBuilder
                    method), 9
setType() (pyDKB.dataflow.communication.stream.StreamBuilder
                   method), 13
source is empty()
                                                                                                          (py-
                   DKB.dataflow.communication.consumer.FileConsumer.FileConsumer
                    method), 8
src_info (pyDKB.dataflow.communication.producer.ProducerBuilder
                    attribute), 9
stop() (pyDKB.dataflow.stage.AbstractStage.AbstractStage
                   method), 19
stop() (pyDKB.dataflow.stage.ProcessorStage method),
stop() (pyDKB.dataflow.stage.ProcessorStage.ProcessorStage
                   method), 21
Stream (class in pyDKB.dataflow.communication.stream),
Stream (class in pyDKB.dataflow.communication.stream.Stream),
                    15
StreamBuilder
                                                  (class
                                                                                  in
                                                                                                           ру-
                   DKB.dataflow.communication.stream), 13
streamClass (pyDKB.dataflow.communication.stream.StreamBuilder
                   attribute), 13
StreamConsumer
                                                     (class
                                                                                    in
                   DKB.dataflow.communication.consumer.StreamConsumer),
StreamException, 16
StreamProducer
                                                    (class
                                                                                   in
                                                                                                           ру-
                    DKB.dataflow.communication.producer.StreamProducer),
subdir() (pyDKB.dataflow.communication.producer.FileProducer.FileProducer
                    method), 10
subdir() (pyDKB.dataflow.communication.producer.HDFSProducer.HDFSProducer
                   method), 11
Т
TTLMessage
                                                 (class
                                                                                 in
                                                                                                           ру-
                   DKB.dataflow.communication.messages),
Type (class in pyDKB.common.Type), 3
typeName() (pyDKB.dataflow.communication.messages.AbstractMessage
                   class method), 16
V
valueByKey() (in module pyDKB.common.json_utils), 6
W
write() (pyDKB.dataflow.communication.producer.Producer.Producer
                    method), 12
write() (pyDKB.dataflow.communication.stream.OutputStream
                   method), 14
```