

OpenEye
SCIENTIFIC

Saiph TK – Java

Release 2026.1.0

Cadence Design Systems, Inc.

August 06, 2026

CONTENTS

1 Examples **3**
- 1.1 Traversing File with an OEMolFileConverter . . . 3
- 1.2 Set Schema Record to Shape OERecord Extraction . . . 3
- 1.3 OERecord to OEMol . . . 3
- 1.4 OEMol to OERecord . . . 3
- 1.5 OERecord to OEDesignUnit . . . 3
- 1.6 File Records to Pandas DataFrame . . . 3

2 Preliminary OESaiph API **5**
- 2.1 Preliminary OESaiph Classes . . . 5
  - 2.1.1 OEMolFileConverter . . . 5
  - 2.1.2 OEMolConversionOptions . . . 6
  - 2.1.3 OESmilesFileConverter . . . 7
  - 2.1.4 OECSVFileConverter . . . 8
  - 2.1.5 OECSVConversionOptions . . . 9
  - 2.1.6 OEDBFileConverter . . . 10
  - 2.1.7 OEDUFileConverter . . . 11
  - 2.1.8 OERecordToCSV . . . 12
  - 2.1.9 OERecordToCSVOptions . . . 12
  - 2.1.10 OEShapeQueryConverter . . . 13
  - 2.1.11 OEConversionOptionsBase . . . 14
  - 2.1.12 ArchiveConverter . . . 15
- 2.2 Preliminary OESaiph Functions . . . 16
  - 2.2.1 GetConverter . . . 16
  - 2.2.2 OEMolToRecord . . . 16
  - 2.2.3 OERecordToDesignUnit . . . 16
  - 2.2.4 OERecordToMol . . . 17
  - 2.2.5 OESaiphGetArch . . . 17
  - 2.2.6 OESaiphGetLicensee . . . 17
  - 2.2.7 OESaiphGetPlatform . . . 17
  - 2.2.8 OESaiphGetRelease . . . 17
  - 2.2.9 OESaiphGetSite . . . 18
  - 2.2.10 OESaiphGetVersion . . . 18
  - 2.2.11 OESaiphIsLicensed . . . 18
  - 2.2.12 read_record_file_to_dataframe . . . 18

3 Release History **19**
- 3.1 Saiph TK 1.0.1 . . . 19
  - 3.1.1 Python-Specific Changes . . . 19
- 3.2 Saiph TK 1.0.0 . . . 19

3.2.1 New Features 19

Index **21**

Attention

This API is currently available in C++ and Python.

Saiph TK provides the functionalities to easily extract, transfer, and load (ETL) files. It has six file-converting classes: *OEMolFileConverter*, *OESmilesFileConverter*, *OECSVFileConverter*, *OEDUFileConverter*, *OEShapeQueryConverter*, and *OERecordToCSV*. *OEMolFileConverter* can be utilized on any file that passes the `OEIsReadable` check. *OESmilesFileConverter* can be utilized on SMI, ISM, CAN, CXSMILES, and USM files. The other converters are specific to their file type. The Python version of the toolkit provides an *ArchiveConverter*, which is an iterator class that converts tarball or zip files into OERecord by returning the best-suited converter iterator for each file in the compressed file.

OEMolFileConverter and *OESmilesFileConverter* both take in *OEMolConversionOptions*, which provides options to clear SDData, set conformer tests, and set and get generic data, as well as allowing the user to set a Mol Title Column. *OECSVFileConverter* takes in *OECSVConversionOptions* for setting the delimiter. The main five converters utilize Open() and GetNextRecord() methods to traverse through a file. Open() is where the file name and converter options can be set. GetNextRecord() populates an OEMolRecord while stepping through the data of the file. This function returns Saiph return codes: `SaiphReturnCode::Success`, `SaiphReturnCode::Fail`, and `SaiphReturnCode::EndOfFile`.

OERecordToCSV directly converts a file into a CSV file. It has *OERecordToCSVOptions* for setting the delimiter.

Saiph TK also provides functions like *`GetConverter`*, *`OEMolToRecord`*, *`OERecordToMol`*, and *`OERecordToDesignUnit`*. *`GetConverter`* returns the best converter for the file, and the other functions convert between OERecord and the respective class type. On the Python side, *`read_record_file_to_dataframe`* give the utility of creating a Pandas DataFrame from any file that passes the `OEIsReadable` check. The user is required to have Pandas installed in their Python environment.

CHAPTER

ONE

EXAMPLES

1.1 Traversing File with an OEMolFileConverter

This is an example of a *OEMolFileConverter* creating a vector of an OERecord from a string file input.

1.2 Set Schema Record to Shape OERecord Extraction

This example sets an OERecord as the schema to the *OEMolConversionOptions*, giving the *OEMolFileConverter* specific OEField to extract from the file given.

1.3 OERecord to OEMol

This example creates an OEMol from the first OERecord produced by the input file.

1.4 OEMol to OERecord

This example creates an OERecord from the OEMol found in the input file.

1.5 OERecord to OEDesignUnit

This example creates an OEDesignUnit from an OERecord.

1.6 File Records to Pandas DataFrame

This example takes in a file and creates a pandas DataFrame from the OERecord produced. The user must have pandas installed in their Python environment to gain access to this function.

CHAPTER

TWO

PRELIMINARY OESAIPH API

2.1 Preliminary OESaiph Classes

2.1.1 OEMolFileConverter

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEMolFileConverter
```

The *OEMolFileConverter* class extracts data from any files that pass the `OEIsReadable` check into OERecords. Custom data extraction based on OEField inclusion (SD and generic data) and the addition of conformer tests can be set up through the *OEMolConversionOptions*.

Constructors

```
OEMolFileConverter()
OEMolFileConverter(const OEMolFileConverter &)
```

Open

```
bool Open(const std::string& path,
          const std::string& displayName = std::string()) override
bool Open(const std::string& path,
          const OEMolConversionOptions& options,
          const std::string& displayName = std::string())
```

Returns a boolean based on whether the file can be opened. *OEMolConversionOptions* is set at open.

Close

```
void Close()
```

Closes the stream.

GetNextRecord

```
unsigned int GetNextRecord(OEDataFlow::OEMolRecord& record) override
```

Steps through each row of the file adding to the OEMolRecord passed into the function. The function returns constants based on each pass. Return codes: `SaiphReturnCode::Success`, `SaiphReturnCode::Fail`, `SaiphReturnCode::EndOfFile`

GetProgress

```
float GetProgress()
```

Returns the percentage of the file that has been traversed.

GetSchemaRecord

```
OEDataFlow::OEMolRecord GetSchemaRecord() const
```

Returns an OEMolRecord schema of the OERecord produced in each *OEMolFileConverter.GetNextRecord* iteration. If a schema record was not given to the *OEMolConversionOptions*, one is produced after running *OEMolFileConverter.Open* with a valid file.

2.1.2 OEMolConversionOptions

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEMolConversionOptions : public OEConversionOptionsBase
```

Gives many factors to customize ETL for the *OEMolFileConverter*. SD data can be cleared, conformer tests can be turned on, custom generic data can be extracted, and custom Mol Title fields are options provided by *OEMolConversionOptions*.

Constructors

```
OEMolConversionOptions()
OEMolConversionOptions(OEMolConversionOptions &&)
OEMolConversionOptions(const OEMolConversionOptions &)
```

GetClearMolData

```
bool GetClearMolData() const
```

Returns a boolean depending on whether the SD data of the OERecord will be cleared.

GetConfTest

```
bool GetConfTest() const
```

Returns a boolean depending on whether the conformer tests are set during extraction.

GetGenericDataTags

```
std::vector<std::string> GetGenericDataTags() const
```

Returns a vector of a string of the generic data tags specified to be extracted from the file.

GetMolTitleField

```
std::string GetMolTitleField() const
```

Returns a string of the OEField name for the Mol Title OEField that will be added. Only adds an OEField if not left blank.

HasGenericDataTags

```
bool HasGenericDataTags() const
```

Returns a boolean depending on whether any generic data tags are set.

SetClearMolData

```
void SetClearMolData(bool state)
```

Boolean for clearing the SD data of the file.

SetConfTest

```
void SetConfTest(bool state)
```

Boolean for turning on conformer tests upon loading the file.

SetGenericDataTags

```
void SetGenericDataTags(const std::vector<std::string>& tags)
```

Sets the generic data tags extracted from the file.

SetMolTitleField

```
void SetMolTitleField(const std::string fieldName)
```

Sets the name of the Mol Title OEField. This OEField is only added if the fieldName is not blank.

2.1.3 OESmilesFileConverter

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OESmilesFileConverter
```

The *OESmilesFileConverter* class extracts data from SMI, ISM, CAN, CXSMILES, and USM files into OERecord. With the addition of an *OEMolConversionOptions*, a SMILES OEField and a molecule title OEField can be added to the OERecord.

Constructors

```
OESmilesFileConverter()
OESmilesFileConverter(const OESmilesFileConverter &)
```

Open

```
bool Open(const std::string& path,
          const std::string& displayName = std::string()) override
bool Open(const std::string& path,
          const OEMolConversionOptions& options,
          const std::string& displayName = std::string())
```

Returns a boolean based on whether the file can be opened. *OEMolConversionOptions* is set at open.

Close

```
void Close()
```

Closes the stream.

GetNextRecord

```
unsigned int GetNextRecord(OEDataFlow::OEMolRecord& record) override
```

Steps through each row of the file adding to the OEMolRecord passed into the function. The function returns constants based on each pass. Return codes: `SaiphReturnCode::Success`, `SaiphReturnCode::Fail`, `SaiphReturnCode::EndOfFile`

GetProgress

```
float GetProgress()
```

Returns the percentage of the file that has been traversed.

2.1.4 OECSVFileConverter

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OECSVFileConverter
```

The *OECSVFileConverter* class extracts data from CSV and TSV files into OERecord. Custom data extraction based on delimiter type and OEField inclusion can be set up through the *OECSVConversionOptions*.

Constructors

```
OECSVFileConverter()
OECSVFileConverter(const OECSVFileConverter &)
```

Open

```
bool Open(const std::string& path,
          const std::string& displayName = std::string()) override
bool Open(const std::string& path,
          const OECSVConversionOptions& options,
          const std::string& displayName = std::string())
```

Returns a boolean based on whether the file can be opened. *OECSVConversionOptions* is set at open.

Close

```
void Close()
```

Closes the stream.

GetNextRecord

```
unsigned int GetNextRecord(OEDataFlow::OEMolRecord& record) override
```

Steps through each row of the file adding to the OEMolRecord passed into the function. The function returns constants based on each pass. Return codes: `SaiphReturnCode::Success`, `SaiphReturnCode::Fail`, `SaiphReturnCode::EndOfFile`

GetProgress

```
float GetProgress()
```

Returns the percentage of the file that has been traversed.

2.1.5 OECSVConversionOptions

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OECSVConversionOptions
```

Stores the delimiter for the *OECSVFileConverter*.

Constructors

```
OECSVConversionOptions()
OECSVConversionOptions(OECSVConversionOptions &&)
OECSVConversionOptions(const OECSVConversionOptions &)
```

Default constructor that initializes an *OECSVConversionOptions* with the delimiter set to a comma.

GetDelimiter

```
std::string GetDelimiter() const
```

Returns the delimiter utilized when parsing a row as a string.

SetDelimiter

```
void SetDelimiter(const std::string delimiter)
```

Sets the delimiter utilized to parse a row.

2.1.6 OEDBFileConverter

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEDBFileConverter
```

The *OEDBFileConverter* class extracts data from OEDB files into OERecord.

Constructors

```
OEDBFileConverter()
OEDBFileConverter(const OEDBFileConverter &)
```

Open

```
bool Open(const std::string& path,
          const std::string& displayName = std::string())
```

Returns a boolean based on whether the file can be opened.

Close

```
void Close()
```

Closes the stream.

GetNextRecord

```
unsigned int GetNextRecord(OEDataFlow::OEMolRecord& record) override
```

Steps through each row of the file adding to the OEMolRecord passed into the function. The function returns constants based on each pass. Return codes: `SaiphReturnCode::Success`, `SaiphReturnCode::Fail`, `SaiphReturnCode::EndOfFile`

GetProgress

```
float GetProgress()
```

Returns the percentage of the file that has been traversed.

2.1.7 OEDUFileConverter

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEDUFileConverter
```

The *OEDUFileConverter* class extracts the OEDesignUnit from an OEDU file into OERecord. The Iridium category is determined by each OERecord extracted.

Constructors

```
OEDUFileConverter()
OEDUFileConverter(const OEDUFileConverter &)
```

Open

```
bool Open(const std::string& path,
          const std::string& displayName = std::string()) override
```

Returns a boolean based on whether the file can be opened.

Close

```
void Close()
```

Closes the stream.

GetNextRecord

```
unsigned int GetNextRecord(OEDataFlow::OEMolRecord& record) override
```

Steps through each row of the file adding to the OEMolRecord passed into the function. The function returns constants based on each pass. Return codes: `SaiphReturnCode::Success`, `SaiphReturnCode::Fail`, `SaiphReturnCode::EndOfFile`

GetProgress

```
float GetProgress()
```

Returns the percentage of the file that has been traversed.

2.1.8 OERecordToCSV

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OERecordToCSV
```

Converts an `OEIsReadable` file into a CSV file. Modifications such as setting the delimiter type, ordering the OEField, and including a molecule title can be set up through the *OERecordToCSVOptions*.

Constructors

```
OERecordToCSV()
```

Open

```
bool Open(const std::string& path,
          const OERecordToCSVOptions& opts = OERecordToCSVOptions())
```

Returns a boolean based on whether the file can be opened. *OERecordToCSVOptions* is set at open.

Write

```
bool Write(const std::string& path)
bool Write(OEPlatform::oeostream& ofs)
```

Writes the CSV to a file path or into a stream.

2.1.9 OERecordToCSVOptions

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OERecordToCSVOptions
```

Stores the delimiter, the order of the OEFields, and whether a molecule title field is included for the *OERecordToCSV*.

Constructors

```
OERecordToCSVOptions()
OERecordToCSVOptions(OERecordToCSVOptions &&)
OERecordToCSVOptions(const OERecordToCSVOptions &)
```

GetDelimiter

```
char GetDelimiter() const
```

Returns the delimiter set for the CSV file.

GetIncludeMolTitle

```
bool GetIncludeMolTitle() const
```

Returns a boolean based on whether a molecule title field will be added.

GetOrderedFields

```
std::vector<std::string> GetOrderedFields() const
```

Returns a vector of strings for how the OEField are ordered.

SetDelimiter

```
void SetDelimiter(char delimiter)
```

Sets the delimiter for the CSV file.

SetIncludeMolTitle

```
void SetIncludeMolTitle(bool state)
```

Boolean for adding a Mol Title OEField to the CSV.

SetOrderedFields

```
void SetOrderedFields(const std::vector<std::string>& orderedFields)
```

Sets the OEField order of the CSV file.

2.1.10 OEShapeQueryConverter

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEShapeQueryConverter
```

The *OEShapeQueryConverter* class extracts the OEShapeQuery from SQ files into OERecord.

Constructors

```
OEShapeQueryConverter()
OEShapeQueryConverter(const OEShapeQueryConverter &)
```

Open

```
bool Open(const std::string& path,
          const std::string& displayName = std::string())
```

Returns a boolean based on whether the file can be opened.

Close

```
void Close()
```

Closes the stream.

GetNextRecord

```
unsigned int GetNextRecord(OEDataFlow::OEMolRecord& record) override
```

Steps through each row of the file adding to the OEMolRecord passed into the function. The function returns constants based on each pass. Return codes: `SaiphReturnCode::Success`, `SaiphReturnCode::Fail`, `SaiphReturnCode::EndOfFile`

GetProgress

```
float GetProgress()
```

Returns the percentage of the file that has been traversed.

2.1.11 OEConversionOptionsBase

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEConversionOptionsBase
```

Constructors

```
OEConversionOptionsBase()
OEConversionOptionsBase(OEConversionOptionsBase &&)
OEConversionOptionsBase(const OEConversionOptionsBase &)
```

GetErrorField

```
std::string GetErrorField() const
```

GetSamplePercent

```
float GetSamplePercent() const
```

GetSchemaLimit

```
unsigned int GetSchemaLimit() const
```

GetSchemaRecord

```
const OEDataFlow::OERecord& GetSchemaRecord() const
```

GetSmilesFieldName

```
std::string GetSmilesFieldName() const
```

GetUniqueValuesLimit

```
unsigned int GetUniqueValuesLimit() const
```

SetErrorField

```
void SetErrorField(const std::string& fieldName)
```

SetSamplePercent

```
void SetSamplePercent(float limit)
```

SetSchemaLimit

```
void SetSchemaLimit(unsigned int limit)
```

SetSchemaRecord

```
void SetSchemaRecord(const OEDataFlow::OERecord& record)
```

SetSmilesFieldName

```
void SetSmilesFieldName(const std::string fieldName)
```

SetUniqueValuesLimit

```
void SetUniqueValuesLimit(unsigned int limit)
```

2.1.12 ArchiveConverter

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python. This functionality is only available in the Python toolkits.

```
class ArchiveConverter
```

ArchiveConverter is an iterator class which converts tarball or zip files into OERecord. This class provides the best-suited converter iterator for each file in the compressed file as it iterates. This provides the utility of combining multiple datasets with the same OERecord setup into one large OERecord or creating multiple OERecord for each file in the compressed file.

Constructors

```
ArchiveConverter(path: str)
```

```
from openeye import oechem, oesaiph

path = "path_to_file.zip"
for convs in ArchiveConverter(path):
   # provides the best-suited converter iterator

   for rec in convs:
      # iterates through each record in the file found in the compressed file
```

2.2 Preliminary OESaiph Functions

2.2.1 GetConverter

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
std::shared_ptr<OEConverterBase>
std::shared_ptr<OEConverterBase> GetConverter(const std::string& filename,
                                              const std::string& displayName =␣
↪std::string())
```

This function returns the best-suited converter to the file type given.

2.2.2 OEMolToRecord

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
OEDataFlow::OERecord OEMolToRecord(const OEChem::OEMolBase& mol)
```

Returns an OERecord object from an OEMolBase object (mol). Other data in the OEMolBase will be set as OEField in the OERecord. This includes OEScalarGrid, OESkewGrid, OESurface, OEDesignUnit, other OEMol, OEFingerPrint, OEShapeQuery, OEBoL, and so on.

2.2.3 OERecordToDesignUnit

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
void OERecordToDesignUnit(OEDataFlow::OERecord& rec,
                          OEBio::OEDesignUnit& du)
```

Takes an existing OERecord object (rec) and generates a new OEDesignUnit object (DU) if an OEDesignUnit OEField is found in the OERecord.

2.2.4 OERecordToMol

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
void OERecordToMol(OEDataFlow::OERecord& rec, OEChem::OEMolBase& mol,
                   bool strict = false, bool fallback = false)
```

Takes an existing OERecord object (rec) and generates a new OEMolBase object (mol) with the data provided in the OERecord as data set to the OEMolBase. If *strict* is set to true, then an error will be thrown instead of a warning when data cannot be set to the OEMolBase. When *fallback* is set true, the function will check the OERecord for any OEMol object if no specific OEMol OEField named *Molecule* is found.

2.2.5 OESaiphGetArch

```
const char* OESaiphGetArch()
```

This function returns the architecture of the current version of Saiph TK. Examples include:

- microsoft-win32-x86
- redhat-RHEL9-x86_64

2.2.6 OESaiphGetLicensee

```
std::string OESaiphGetLicensee()
bool OESaiphGetLicensee(std::string& licensee)
```

This function returns the LICENSEE from the current valid Saiph TK license.

2.2.7 OESaiphGetPlatform

```
const char* OESaiphGetPlatform()
```

This function returns the platform, including compiler version, of the current version of Saiph TK. Examples include:

- microsoft-win32-msvc9-MD-x86
- redhat-RHEL9-g++11.11-x86_64

2.2.8 OESaiphGetRelease

```
const char* OESaiphGetRelease()
```

This function returns the version of this toolkit release. For example: `1.0.0`.

2.2.9 OESaiphGetSite

```
std::string OESaiphGetSite()
bool OESaiphGetSite(std::string& site)
```

This function returns the SITE from the current valid Saiph TK license.

2.2.10 OESaiphGetVersion

```
unsigned int OESaiphGetVersion()
```

This function returns the build date of the toolkit as an unsigned int. For example: `20240227`.

2.2.11 OESaiphIsLicensed

```
bool OESaiphIsLicensed(const char* feature = 0,
                       unsigned int* expdate = 0)
```

This function determines whether a valid license file is present. This function may be called without a legitimate run-time license to determine whether it is safe to call any **Saiph TK** functionality.

The 'feature' argument can be used to check for a valid license to **Saiph TK** along with the following features: 'python'. Pass `nullptr` (or `0`) to check the default **Saiph TK** feature.

The 'expdate' argument can be used to obtain the expiration date of the current **Saiph TK** license. If non-null, the expiration date is written as a `YYYYMMDD` integer. Pass `nullptr` to ignore.

See also

- `OEChemIsLicensed` function

2.2.12 read_record_file_to_dataframe

Attention

This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python. This functionality is only available in the Python toolkits.

```
def read_record_file_to_dataframe(path: str, serializable: bool = False,
                                  pandasCheck: bool = True) -> Union["Series", "DataFrame
↪"]
```

Creates a pandas DataFrame from any file that passes the `OEIsReadable` check. The user is required to have pandas installed in their Python environment. If *serializable* is set true, then the DataFrame will contain all non-POD types as bytes.

CHAPTER

THREE

RELEASE HISTORY

3.1 Saiph TK 1.0.1

3.1.1 Python-Specific Changes

- The following APIs are now properly wrapped in Python:
  - *OESaiphGetArch*
  - *OESaiphGetLicensee*
  - *OESaiphGetPlatform*
  - *OESaiphGetRelease*
  - *OESaiphGetSite*
  - *OESaiphGetVersion*
  - *OESaiphIsLicensed*

3.2 Saiph TK 1.0.0

This is the first release of **Saiph TK** in C++ and Python.

3.2.1 New Features

- *OEMolFileConverter*, *OESmilesFileConverter*, *OECSVFileConverter*, *OEDUFileConverter*, *OEShapeQueryConverter*, *OERecordToCSV*, and *ArchiveConverter* were added to enhance efficient ETL with the OpenEye toolkits.
- *OEMolConversionOptions* and *OECSVConversionOptions* are classes added to utilize *OEMolFileConverter*, *OESmilesFileConverter*, and *OECSVFileConverter* to help manage the ETL process.
- The function `GetConverter` returns the converter best suited for a file type.
- The function `OEMolToRecord` converts an OEMolBase to an OERecord.
- The function `OERecordToMol` converts an OERecord to an OEMolBase.
- The function `OERecordToDesignUnit` converts an OERecord to an OEDesignUnit.
- The function `read_record_file_to_dataframe` converts any file that passes the `OEIsReadable` check into a pandas DataFrame (User must have pandas installed).

INDEX

C

Close
    OESaiph::OECSVFileConverter, 9
    OESaiph::OEDBFileConverter, 10
    OESaiph::OEDUFileConverter, 11
    OESaiph::OEMolFileConverter, 5
    OESaiph::OEShapeQueryConverter, 13
    OESaiph::OESmilesFileConverter, 8
Constructors
    OESaiph::ArchiveConverter, 15
    OESaiph::OEConversionOptionsBase, 14
    OESaiph::OECSVConversionOptions, 9
    OESaiph::OECSVFileConverter, 8
    OESaiph::OEDBFileConverter, 10
    OESaiph::OEDUFileConverter, 11
    OESaiph::OEMolConversionOptions, 6
    OESaiph::OEMolFileConverter, 5
    OESaiph::OERecordToCSV, 12
    OESaiph::OERecordToCSVOptions, 12
    OESaiph::OEShapeQueryConverter, 13
    OESaiph::OESmilesFileConverter, 8

G

GetClearMolData
    OESaiph::OEMolConversionOptions, 6
GetConfTest
    OESaiph::OEMolConversionOptions, 6
GetDelimiter
    OESaiph::OECSVConversionOptions, 9
    OESaiph::OERecordToCSVOptions, 12
GetErrorField
    OESaiph::OEConversionOptionsBase, 14
GetGenericDataTags
    OESaiph::OEMolConversionOptions, 6
GetIncludeMolTitle
    OESaiph::OERecordToCSVOptions, 12
GetMolTitleField
    OESaiph::OEMolConversionOptions, 7
GetNextRecord
    OESaiph::OECSVFileConverter, 9
    OESaiph::OEDBFileConverter, 10
    OESaiph::OEDUFileConverter, 11
    OESaiph::OEMolFileConverter, 5
    OESaiph::OEShapeQueryConverter, 14
    OESaiph::OESmilesFileConverter, 8
GetOrderedFields
    OESaiph::OERecordToCSVOptions, 13
GetProgress
    OESaiph::OECSVFileConverter, 9
    OESaiph::OEDBFileConverter, 10
    OESaiph::OEDUFileConverter, 11
    OESaiph::OEMolFileConverter, 6
    OESaiph::OEShapeQueryConverter, 14
    OESaiph::OESmilesFileConverter, 8
GetSamplePercent
    OESaiph::OEConversionOptionsBase, 14
GetSchemaLimit
    OESaiph::OEConversionOptionsBase, 14
GetSchemaRecord
    OESaiph::OEConversionOptionsBase, 14
    OESaiph::OEMolFileConverter, 6
GetSmilesFieldName
    OESaiph::OEConversionOptionsBase, 15
GetUniqueValuesLimit
    OESaiph::OEConversionOptionsBase, 15

H

HasGenericDataTags
    OESaiph::OEMolConversionOptions, 7

O

OESaiph::ArchiveConverter, 15
    Constructors, 15
OESaiph::GetConverter, 16
OESaiph::OEConversionOptionsBase, 14
    Constructors, 14
    GetErrorField, 14
    GetSamplePercent, 14
    GetSchemaLimit, 14
    GetSchemaRecord, 14
    GetSmilesFieldName, 15
    GetUniqueValuesLimit, 15
    SetErrorField, 15
    SetSamplePercent, 15

SetSchemaLimit, 15
SetSchemaRecord, 15
SetSmilesFieldName, 15
SetUniqueValuesLimit, 15
OESaiph::OECSVConversionOptions, 9
Constructors, 9
GetDelimiter, 9
SetDelimiter, 10
OESaiph::OECSVFileConverter, 8
Close, 9
Constructors, 8
GetNextRecord, 9
GetProgress, 9
Open, 9
OESaiph::OEDBFileConverter, 10
Close, 10
Constructors, 10
GetNextRecord, 10
GetProgress, 10
Open, 10
OESaiph::OEDUFileConverter, 11
Close, 11
Constructors, 11
GetNextRecord, 11
GetProgress, 11
Open, 11
OESaiph::OEMolConversionOptions, 6
Constructors, 6
GetClearMolData, 6
GetConfTest, 6
GetGenericDataTags, 6
GetMolTitleField, 7
HasGenericDataTags, 7
SetClearMolData, 7
SetConfTest, 7
SetGenericDataTags, 7
SetMolTitleField, 7
OESaiph::OEMolFileConverter, 5
Close, 5
Constructors, 5
GetNextRecord, 5
GetProgress, 6
GetSchemaRecord, 6
Open, 5
OESaiph::OEMolToRecord, 16
OESaiph::OERecordToCSV, 11
Constructors, 12
Open, 12
Write, 12
OESaiph::OERecordToCSVOptions, 12
Constructors, 12
GetDelimiter, 12
GetIncludeMolTitle, 12
GetOrderedFields, 13
SetDelimiter, 13
SetIncludeMolTitle, 13
SetOrderedFields, 13
OESaiph::OERecordToDesignUnit, 16
OESaiph::OERecordToMol, 17
OESaiph::OESaiphGetArch, 17
OESaiph::OESaiphGetLicensee, 17
OESaiph::OESaiphGetPlatform, 17
OESaiph::OESaiphGetRelease, 17
OESaiph::OESaiphGetSite, 17
OESaiph::OESaiphGetVersion, 18
OESaiph::OESaiphIsLicensed, 18
OESaiph::OEShapeQueryConverter, 13
Close, 13
Constructors, 13
GetNextRecord, 14
GetProgress, 14
Open, 13
OESaiph::OESmilesFileConverter, 7
Close, 8
Constructors, 8
GetNextRecord, 8
GetProgress, 8
Open, 8
OESaiph::read_record_file_to_dataframe, 18
Open
OESaiph::OECSVFileConverter, 9
OESaiph::OEDBFileConverter, 10
OESaiph::OEDUFileConverter, 11
OESaiph::OEMolFileConverter, 5
OESaiph::OERecordToCSV, 12
OESaiph::OEShapeQueryConverter, 13
OESaiph::OESmilesFileConverter, 8

S

SetClearMolData
OESaiph::OEMolConversionOptions, 7
SetConfTest
OESaiph::OEMolConversionOptions, 7
SetDelimiter
OESaiph::OECSVConversionOptions, 10
OESaiph::OERecordToCSVOptions, 13
SetErrorField
OESaiph::OEConversionOptionsBase, 15
SetGenericDataTags
OESaiph::OEMolConversionOptions, 7
SetIncludeMolTitle
OESaiph::OERecordToCSVOptions, 13
SetMolTitleField
OESaiph::OEMolConversionOptions, 7
SetOrderedFields
OESaiph::OERecordToCSVOptions, 13
SetSamplePercent
OESaiph::OEConversionOptionsBase, 15

SetSchemaLimit
    OESaiph::OEConversionOptionsBase, 15
SetSchemaRecord
    OESaiph::OEConversionOptionsBase, 15
SetSmilesFieldName
    OESaiph::OEConversionOptionsBase, 15
SetUniqueValuesLimit
    OESaiph::OEConversionOptionsBase, 15

W

Write
    OESaiph::OERecordToCSV, 12