onec_dtools Documentation

Выпуск 0.3.0

infactum

13 February 2016

Оглавление

onec_dtools - это python модуль для работы с бинарными файлами 1С:Предприятие 8 без использования технологической платформы.

Основные возможности:

1. Чтение содержимого файловой базы данных (*.1CD).
2. Распаковка и упаковка файлов-контейнеров (*.cf, *.cfu, *.cfe, *.epf, *.ert, *.hbk).

Содержание

1.1 Установка

1.1.1 PIP

Последняя стабильная версия всегда доступна через pip. Устанавливаем через консоль:

```
$ pip install onec_dtools
```

1.1.2 Github

onec_dtools разрабатывается на Github. Там вы всегда можете найти последнюю develop версию, содержащую данную документацию и тесты. Для получения модуля необходимо склонировать репозиторий и выполнить команду установки:

```
$ git clone https://github.com/Infactum/onec_dtools
$ python setup.py install
```

1.2 Использование

1.2.1 Работа с файлами БД

Для чтения БД используется классс `DatabaseReader`. При инициализации класса считывается основная информация о БД (версия формата, язык и т.д) и список таблиц. Каждая таблица представляет собой объект класса `Table`. Обращение к строкам таблицы может выполняться путем итерирования объекта таблицы, либо путем обращения по индексу. Каждая строка таблицы БД представляет собой объект класса `Row`. Методы работы со строками аналогичны методам работы с таблицами БД.

Стоит обратить внимание на то, что преобразование значений полей из внутреннего формата 1С происходит при обращении к полю. В дальнейшем значение кэшируется внутри объект. Таким образом, чтобы не снижать скорость работы, не рекоммендуется применять методы `Row.as_dict` и `Row.as_list` если не требуются значения всех полей.

Значения полей неограниченной длины представлены объектами класса `Blob`. Значение поля может быть считано в память целиком путем обращения к свойству `Blob.value`. Если объект слишком большой, чтобы поместиться в памяти (размер можно получить через `len(Blob)`), то он может быть считан частями по 256 байт путем итерирования.

Следующий пример демонстрирует чтение данных о пользователях (а так же расшифровку хэшей паролей) из таблицы V8USERS файловой БД.

```
import binascii
import re
import base64
import argparse
import onec_dtools


def extract_hashes(text):
    """
    Получает SHA1 хэши паролей пользователей из расшифрованных данных поля DATA

    :param text: расшифрованное поле DATA
    :return: кортеж хэшей: (SHA1(pwd), SHA1(TO_UPPER(pwd))
    """
    result = re.search('\d+,\d+,"(\S+)","(\S+)",\d+,\d+', text)
    if result is None:
        return
    return tuple([''.join('{:02x}'.format(byte) for byte in base64.decodebytes(x.encode())) for x in result.groups(
)


def decode_data_fld(buffer):
    """
    Декодирование поля DATA таблицы V8USERS

    :param buffer: зашифрованные данные
    :return:
    """
    # Первый байт содержит длину маски шифрования
    # Далее каждая порция байт соответствующей длины поксорена на маску
    mask_length = int(buffer[0])
    j = 1
    decoded = []
    for i in buffer[mask_length + 1:]:
        decoded.append('{:02X}'.format(int(buffer[j] ^ int(i))))
        j += 1
        if j > mask_length:
            j = 1
    decoded_hex_str = ''.join(decoded)
    decoded_bin_str = binascii.unhexlify(decoded_hex_str)
    return decoded_bin_str.decode("utf-8-sig")


if __name__ == '__main__':
    parser = argparse.ArgumentParser()
    parser.add_argument('path_to_1CD', type=str)
    args = parser.parse_args()
    with open(args.path_to_1CD, 'rb') as f:
        db = onec_dtools.DatabaseReader(f)

        print("+{}+{}+{}+{}+".format(6*'-', 50*'-', 42*'-', 42*'-'))
        print("|{:6}|{:50}|{:42}|{:42}|".format('Admin', 'Имя пользователя', 'SHA1', 'SHA1'))
        print("+{}+{}+{}+{}+".format(6*'-', 50*'-', 42*'-', 42*'-'))

        for row in db.tables['V8USERS']:
            if row.is_empty:
                continue
```

```
            hashes = extract_hashes(decode_data_fld(row['DATA'].value))
            if hashes is None:
                continue
            print("|{0[ADMROLE]!r:6}|{0[NAME]:50}|{1[0]:42}|{1[1]:42}|".format(row, hashes))

        print("+{}+{}+{}+{}+".format(6*'-', 50*'-', 42*'-', 42*'-'))
```

Результатом на примере демо базы конфигурации Управляемое приложение будет следующая таблица:

```
+------+--------------------------------------------------+------------------------------------------+-------------
|Админ |Имя пользователя                                  |SHA1                                      |SHA1
+------+--------------------------------------------------+------------------------------------------+-------------
|True  |Администратор                                     |da39a3ee5e6b4b0d3255bfef95601890afd80709  |da39a3ee5e6b4
|False |Менеджер по закупкам                              |da39a3ee5e6b4b0d3255bfef95601890afd80709  |da39a3ee5e6b4
|False |Менеджер по продажам                              |da39a3ee5e6b4b0d3255bfef95601890afd80709  |da39a3ee5e6b4
|False |Продавец                                          |da39a3ee5e6b4b0d3255bfef95601890afd80709  |da39a3ee5e6b4
+------+--------------------------------------------------+------------------------------------------+-------------
```

1.2.2 Работа с контейнерами

Работать с контейнерами можно как используя классы `ContainerReader` и `ContainerWriter` для распаковки/упаковки контейнеров соответственно, так и применяя синтаксический сахар в виде функций `parse` и `build`.

Следующий код реализует возможности распаковки и обратной сборки контейнеров по аналогии с тем, как это делает C++ версия v8unpack:

```
import argparse
import sys
import onec_dtools


def main():
    parser = argparse.ArgumentParser()
    group = parser.add_mutually_exclusive_group(required=True)
    group.add_argument('-P', '--parse', nargs=2, metavar=('in_filename', 'out_dir_name'))
    group.add_argument('-B', '--build', nargs=2, metavar=('in_dir_name', 'out_filename'))

    if len(sys.argv) == 1:
        parser.print_help()
        return 1

    args = parser.parse_args()

    if args.parse is not None:
        onec_dtools.extract(*args.parse)

    if args.build is not None:
        onec_dtools.build(*args.build)


if __name__ == '__main__':
    sys.exit(main())
```

1.3 Описание модулей

1.3.1 database_reader

class onec_dtools.database_reader.DatabaseReader(*db_file*)

Параметры db_file (*BufferedReader*) – файл базы данных

locale = **None**
Язык БД

tables = **None**
Словарь таблиц БД.

Ключ: Имя таблицы

Значение: Объект класса **Table**

total_pages = **None**
Количество страниц в БД

version = **None**
Версия формата

class onec_dtools.database_reader.Table(*db_file*, *description*)
Таблица файловой БД

Параметры

- db_file (*BufferedReader*) – Объект файла БД
- description (*string*) – Описание таблицы во внутреннем формате 1С

__getitem__(*key*)
Реализует интерфейс работы с таблицой как со списком

Параметры key (*int*) – индекс строки

Результат строка таблицы

Тип результата *Row*

__iter__()
Реализует интерфейс перебора строк табилцы

Результат Итератор строк таблицы

__len__()
Позволяет получать число строк в таблице

Результат Общее количество строк в таблице (включая пустые)

Тип результата int

fields = **None**
Словарь описаний полей таблицы

name = **None**
Имя таблицы

class onec_dtools.database_reader.Row(*db_file*, *row_bytes*, *table*)
Строка БД

Параметры

- db_file (*BufferedReader*) – Объект файла БД
- row_bytes (*bytearray*) – Внутреннее представление строки
- table (Table) – Таблица БД, которой принадлежит строка.

__getitem__(*key*)
Позволяет получать значения полей по имени колонки

Параметры key (*string*) – Имя колонки

Результат Значение поля

as_dict(*read_blobs=False*)
Возвращает представление строки таблицы в виде словаря

Параметры read_blobs (*bool*) – Флаг считывания значений BLOB полей

Результат Строка таблицы

Тип результата OrderedDict

as_list(*read_blobs=False*)
Возвращает представление строки таблицы в виде списка

Параметры read_blobs (*bool*) – Флаг считывания значений BLOB полей

Результат Строка таблицы

Тип результата list

is_empty = **None**
Флаг пустой строки. Все поля пустой строки равны None

class onec_dtools.database_reader.Blob(*db_file*, *blob_size*, *blob_offset*, *blob_chunk_offset*, *field_type*)
Поле неограниченной длины

Параметры

- db_file (*BufferedReader*) – Объект файла БД
- blob_size (*int*) – Размер BLOB в байтах
- blob_offset (*int*) – Смещение объекта BLOB данных таблицы в файле БД (страниц)
- blob_chunk_offset (*int*) – Смещение данных внутри BLOB объекта (число блоков по 256 байт)
- field_type (*string*) – тип поля неограниченной длины (I или NT)

__iter__()
Позволяет считывать данные поля блоками.

Результат Итератор BLOB кусками по 256 байт

Тип результата bytearray

__len__()

Результат Размер поля в байтах

Тип результата int

value

Результат Значение поля

Тип результата bytearray или string

class onec_dtools.database_reader.DBObject(*db_file*, *object_offset*)
Объект БД

Параметры

- db_file (*BufferedReader*) – Объект файла БД
- object_offset (*int*) – смещение объекта БД относительно начала файла БД (в страницах)

__len__()
Реализует интерфейс получения размера объета

Результат Размер объекта в байтах

Тип результата int

read(*size=-1*)
Читает не более size байт данных объекта БД

Параметры size (*int*) – Размер считываемых данных. Size < 0 для чтения всего объекта.

Результат данные объекта

Тип результата bytearray

seek(*pos*)
Позиционируется на смещении относительно начала данных объекта

Параметры pos (*int*) – Байт от начала данных объекта

class onec_dtools.database_reader.FieldDescription
Описание поля таблицы

type

null_exists

length

precision
Длина дробной части для типа Numeric

case_sensitive

data_offset
Смещение данных поля относительно начала строки (байт)

data_length
Длина данных поля (байт)

onec_dtools.database_reader.database_header(*db_file*)
Читает заголовок файла БД

Параметры db_file (*BufferedReader*) – Объект файла БД

Результат версия и число страниц

Тип результата tuple

onec_dtools.database_reader.root_object(*db_file*)
Читает корневой объет БД

Параметры db_file (*BufferedReader*) – Объект файла БД

Результат язык и смещения объектов описания таблиц БД

Тип результата tuple

onec_dtools.database_reader.raw_tables_descriptions(*db_file*, *tables_offsets*)
Получает описания таблиц БД во внутренне формате 1С.

Параметры

- db_file (*BufferedReader*) – Объект файла БД
- tables_offsets (*tuple*) – Смещения объектов описания таблиц БД

Результат Описания таблиц

Тип результата list

onec_dtools.database_reader.calc_field_size(*field_type*, *length*)
Рассчитывает размер данных поля

Параметры

- field_type (*string*) – Тип поля
- length (*int*) – Длина поля

Результат Длина поля в байтах

Тип результата int

onec_dtools.database_reader.numeric_to_int(*numeric*, *length*, *precision*)
Преобразуем Numeric формат 1С в число.

Параметры

- numeric (*bytearray*) – число в формате Numeric
- length (*int*) – длина поля
- precision (*int*) – точность

Результат Числовое представление

Тип результата int или float

onec_dtools.database_reader.nvc_to_string(*nvc*)
Преобразует NVarChar формат 1С в строку.

Параметры nvc (*bytearray*) – строка в формате NVC

Результат Строковое представление

Тип результата string

onec_dtools.database_reader.bytes_to_datetime(*bts*)
Пробразует данные типа DT в дату/время

Параметры bts (*bytearray*) – значение в формате DT

Результат дата+время

Тип результата datetime

1.3.2 container_reader

class onec_dtools.container_reader.ContainerReader(*file*)
Класс для чтения контейнеров

entries = **None**
Список файлов в контейнере

extract(*path*, *deflate=False*, *recursive=False*)
Распаковывает содержимое контейнера в каталог

Параметры

- path (*string*) – каталог распаковки
- deflate (*bool*) – разархивировать содержимое файлов
- recursive (*bool*) – выполнять рекурсивно

onec_dtools.container_reader.extract(*filename*, *folder*)
Распаковка контейнера. Сахар для ContainerReader

Параметры

- filename (*string*) – полное имя файла-контейнера
- folder (*string*) – каталог назначения

onec_dtools.container_reader.read_header(*file*)
Считывает заголовок контейнера.

Параметры file (*BufferedReader*) – объект файла контейнера

Результат Заголовок контейнера

Тип результата Header

onec_dtools.container_reader.read_block(*file*, *offset*, *max_data_length=None*)
Считывает блок данных из контейнера.

Параметры

- file (*BufferedReader*) – объект файла контейнера
- offset (*int*) – смещение блока в файле контейнера (байт)
- max_data_length (*int*) – максимальный размер считываемых данных из блока (байт)

Результат объект блока данных

Тип результата Block

onec_dtools.container_reader.read_document(*file*, *offset*)
Считывает документ из контейнера. В качестве данных документа возвращается генератор.

Параметры

- file (*BufferedReader*) – объект файла контейнера
- offset (*int*) – смещение документа в контейнере

Результат объект документа

Тип результата Document

onec_dtools.container_reader.read_full_document(*file*, *offset*)
Считывает документ из контейнера. Данные документа считываются целиком.

Параметры

- `file` (*BufferedReader*) – объект файла контейнера
- `offset` (*int*) – смещение документа в контейнере (байт)

Результат объект документа

Тип результата Document

`onec_dtools.container_reader.parse_datetime(time)`
Преобразует внутренний формат хранения дат файлов в контейнере в обычную дату

Параметры `time` (*string*) – внутреннее представление даты

Результат дата/время

Тип результата datetime

`onec_dtools.container_reader.read_entries(file)`
Считывает оглавление контейнера

Параметры `file` (*BufferedReader*) – объект файла контейнера

Результат словарь файлов в контейнере

Тип результата OrderedDict

1.3.3 container_writer

class `onec_dtools.container_writer.ContainerWriter(file)`
Класс для записи контейнеров

Параметры `file` (*BufferedReader*) – объект файла контейнера

`__enter__()`
Вход в блок. Позволяет применять оператор with.

`__exit__(exc_type, exc_val, exc_tb)`
Выход из блока. Позволяет применять оператор with.

`add_file(fd, name, inflate=False)`
Добавляет файл в контейнер

Параметры

- `fd` (*BufferedReader*) – file-like объект файла
- `name` (*string*) – Имя файла в контейнере
- `inflate` (*bool*) – флаг сжатия

`write_block(data, **kwargs)`
Записывает блок данных в контейнер

Параметры

- `data` – file-like объект
- `kwargs` – Опциональные параметры

Результат смещение записанных данных (байт)

Тип результата int

`write_header()`
Записывает заголовок контейнера

write_toc()
Записывает оглавление контейнера

onec_dtools.container_writer.build(*folder*, *filename*)
Запакоывает каталог в контейнер включая вложенные каталоги. Сахар для ContainerWriter.

Параметры

- folder (*string*) – каталог с данными, запаковываемыми в контейнер
- filename (*string*) – имя файла контейнера

onec_dtools.container_writer.add_entries(*container*, *folder*, *nested=False*)
Рекурсивно добавляет файлы из директории в контейнер

Параметры

- container (*BufferedReader*) – объет файла контейнера
- folder (*string*) – каталог файлов, которые надо поместить в контейнер
- nested (*bool*) – обрабатывать вложенные каталоги

onec_dtools.container_writer.epoch2int(*epoch_time*)
Преобразует время в формате "количество секунд с начала эпохи" в количество сотых микросекундных интервалов с 0001.01.01

Параметры epoch_time (*real*) – время в формате Python

Результат количество сотых микросекундных интервалов

Тип результата int

onec_dtools.container_writer.int2hex(*value*)
Получает строковое представление целого числа в шестнадцатиричном формате длиной не менее 4 байт

Параметры value (*int*) – конвертируемое число

Результат предоставление числа

Type string

onec_dtools.container_writer.get_size(*file*)
Возвращает размер file-like объекта

Параметры file (*BufferedReader*) – объекта файла

Результат размер в байтах

Тип результата int

1.4 История версий

1.4.1 0.3.0

- Рефакторинг механизмы чтения формата 1CD
- Новый API работы с файловой базой
- Значительно улучшена документация

1.4.2 0.2.0

- Исправлена ошибка преобразования значений типа Numeric
- Исправлена ошибка чтения значений полей, допускающих NULL
- Ускорено чтение информации о страницах размещения объектов БД
- Ускорен разбор описаний таблиц БД
- Ускорено преобразование полей типа DateTime
- Преобразование значение в полях таблиц теперь происходит в момент обращения к ним, а не в момент чтения строки

1.4.3 0.1.1

- Исправление ошибок

1.4.4 0.1.0

- Добавлен функционал работы с контейнерами (cf, epf, ert и т.д.)

1.4.5 0.0.3

- Поддержка Python 3.4

1.4.6 0.0.1

- Первая публичная версия
- Реализована поддержка чтения формата 1CD

Сайты onec_dtools

- Репозиторий на github: https://github.com/Infactum/onec_dtools
- Темы на Infostart:
  - http://infostart.ru/public/418553/
  - http://infostart.ru/public/412475/
- Страница проекта в PyPI: https://pypi.python.org/pypi/onec_dtools/

Symbols

A

B

C

D

E

F

G