edipy

Release 0.0.2

Alexandre Barbieri (fakeezz)

Jul 11, 2019

CONTENTS:

CHAPTER

ONE

INTRODUCTION

edipy came to help you to parse positional data easily. Just with a class declaration you can read texts and files.

1.1 Basic usage

With **edipy** you just need declare a class using fields

```
from datetime import date
from edipy import fields, parser

class Example(fields.EDIModel):
    name = fields.String(5)
    date = fields.Date(8, '%Y%m%d')
    description = fields.String(7)
    likes = fields.Integer(4)

data = 'EDIPY20190719AWESOME9999'
example = parser.parse(Example, data)

assert example.name == 'EDIPY'
assert example.date ==  date(2019, 7, 19)
assert example.description == 'AWESOME'
assert example.likes == 9999
```

1.2 Installation

You just need to install `edipy` library:

```
pip install edipy
```

1.3 Complex models

If you need to read data according an specific format, such as ANSI X12, you can create composed types.

```
from edipy import fields, parser

class ISASegment(fields.EDIModel):
```

(continues on next page)

(continued from previous page)

```
    identifier = fields.Identifier("ISA")
    content = fields.String(5)

class GSSegment(fields.EDIModel):
    identifier = fields.Identifier("GS")
    content = fields.String(5)

class ANSIX12(fields.EDIModel):
    isa = fields.Register(ISASegment, occurrences=1)
    gs = fields.Register(GSSegment, occurrences=2)

data = 'ISA*100*\r\nGS*200*GS*300*'
ansi = parser.parse(ANSIX12, data)

assert ansi.isa.identifier == 'ISA'
assert ansi.isa.content == '*100*'

assert len(ansi.gs) == 2
assert ansi.gs[0].identifier == 'GS'
assert ansi.gs[0].content == '*200*'
assert ansi.gs[1].content == '*300*'
```

1.4 Validators

There are basic validators that you can use or extend to check if data is correct.

```
from edipy import fields, parser, validators, exceptions

class User(fields.EDIModel):
    name = fields.String(10)
    age = fields.Integer(2, required=False, validators=[validators.MinValue(18)])
    email = fields.String(20, required=False, validators=[validators.Email()])

try:
    data = 'Someone   17someone@net.com  '
    invalid_age = parser.parse(User, data)
except exceptions.ValidationError as e:
    print("MinValue: {}".format(e.message))

try:
    data = 'Someone   19someoneanet.com  '
    invalid_email = parser.parse(User, data)
except exceptions.ValidationError as e:
    print("Email: {}".format(e.message))
```

CHAPTER

TWO

FIELDS

Define which type, size and rules of validation for specific part of the data. There are many built-in fields but you can extend easily.

2.1 Built-in Fields

class edipy.fields.**String**(*size*, *required=True*, *validators=None*)

class edipy.fields.**Integer**(*size*, *zfill=False*, *required=True*, *validators=None*)

class edipy.fields.**Identifier**(*identifier*, *required=True*, *validators=None*)

class edipy.fields.**Decimal**(*size*, *digits=0*, *required=True*, *validators=None*)

class edipy.fields.**DateTime**(*size*, *date_format*, *required=True*, *validators=None*)

class edipy.fields.**Date**(*size*, *date_format*, *required=True*, *validators=None*)

class edipy.fields.**Time**(*size*, *date_format*, *required=True*, *validators=None*)

class edipy.fields.**CompositeField**(*cls*, *occurrences=1*, *required=True*)

class edipy.fields.**Register**(*cls*, *occurrences=1*, *required=True*)

class edipy.fields.**Enum**(*values*, *required=True*, *validators=None*)

CHAPTER

THREE

VALIDATORS

A validator takes a value and raise `ValidationError` if it doesn‘t meet some criteria.

3.1 Built-in validators

class edipy.validators.**Range**(*min_value*, *max_value*)
Validate if a value is within a specific range of values

Parameters

- **min_value** – minimum value of the validation range.
- **max_value** – maximum value of the validation range.

class edipy.validators.**MaxValue**(*max_value*)
Validate if a value is greater than the limit

Parameters **max_value** – maximum value allowed.

class edipy.validators.**MinValue**(*min_value*)
Validate if a value is less than the limit

Parameters **min_value** – minimum value allowed.

class edipy.validators.**Regex**(*pattern*)
Validates if a value matches the pattern

Parameters **pattern** – regular expression

class edipy.validators.**Email**
Validates if value is a valid email

3.2 How to extend

```
from edipy import validators, fields, exceptions, parser

class MyValidator(validators.Validator):

    def validate(self, value):
        if value != "edi":
            raise exceptions.ValidationError(message=u"Value should be edi")
        return True
```

(continues on next page)

(continued from previous page)

```
class ValidatorExample(fields.EDIModel):
    data = fields.String(3, validators=[MyValidator()])

try:
    data = 'aaa'
    example = parser.parse(ValidatorExample, data)
except exceptions.ValidationError as e:
    print(e.message)
```

CHAPTER

FOUR

CHANGELOG

You can see all changes at CHANGELOG

CHAPTER

FIVE

INDICES AND TABLES

- search

INDEX