enumagic

Release 0.1.1

Aug 15, 2022

Contents

Python enums that work like magic.

class enumagic.**IterEnum**

Bases: enum.Enum

Enum class that can be used as an iterable.

__class__

alias of *enumagic.IterMeta*

Examples

```
>>> class IterExample(IterEnum):
...     A = 'Alice'
...     B = 'Bob'
>>> dict(IterExample)
{'A': 'Alice', 'B': 'Bob'}
```

class enumagic.**IterMeta**

Bases: enum.EnumMeta

Iterable enum metaclass.

__contains__(*item: Any*) → bool

Check whether the enum contains a certain item

Parameters **item** (Any) – A string or enum instance.

Returns bool – True if the enum has a member that matches the given item, False otherwise.

Raises TypeError – If the item is not a string or enum instance.

Examples

```
>>> 'B' in IterExample
True
>>> 'C' in IterExample
False
```

__iter__() → Iterator[Tuple[str, _VT]]

Iterate over the entries of the enum.

Yields tuple of str, object – The next tuple where the first element is the name of the enum instance and the second element is the value of the enum instance.

Examples

```
>>> it = iter(IterExample)
>>> next(it)
('A', 'Alice')
```

class enumagic.**MappingEnum**(*index: int*, *label: str*)

Bases: enum.Enum

Enum class which maps labels to indices.

Variables

- **index** (*int*) – An integer that will be used as the index.
- **label** (*str*) – A string that will be used as the label.

Examples

```
>>> class MappingExample(MappingEnum):
...     A = 0, 'Alice'
...     B = 1, 'Bob'
>>> '%d, %s' % (MappingExample.B.index, Example.B.label)
'1, Bob'
```

__class__
alias of *enumagic.MappingMeta*

__str__() → str
Return the instance as a string.

Returns str – The label of the instance.

Examples

```
>>> str(MappingExample.A)
'Alice'
```

__index__() → int
Return the instance as an index.

Returns int – The index of the instance.

Examples

```
>>> test = ['first', 'second']
>>> test[MappingExample.B]
'second'
```

__int__() → int
Return the instance as an integer.

Returns int – The index of the instance.

Examples

```
>>> int(MappingExample.A)
0
```

class enumagic.**MappingMeta**
Bases: enum.EnumMeta

Mapping enum metaclass.

__call__(*value: Any*) → _ET
Get an enum instance from the given value.

Parameters **value** (Any) – The value to look for in the members of the enum.

Returns Enum – An enum instance that corresponds to the value.

Raises ValueError – If the given value is invalid.

Examples

```
>>> MappingExample(0)
<MappingExample.A: (0, 'Alice')>
>>> MappingExample('Bob')
<MappingExample.B: (1, 'Bob')>
```

__iter__() → Iterator[Tuple[int, str]]
Iterate over the values of the enum.

Yields tuple of int, str – The next tuple where the first element is the index of the enum instance and the second element is the label of the enum instance.

Examples

```
>>> list(MappingExample)
[(0, 'Alice'), (1, 'Bob')]
```

indices
Get the indices of the enum.

Type tuple of int

Examples

```
>>> MappingExample.indices
(0, 1)
```

items
Get a mapping of label/index pairs.

Type dict of str to int

Examples

```
>>> MappingExample.items
{'Alice': 0, 'Bob': 1}
```

labels
Get the labels of the enum.

Type tuple of str

Examples

```
>>> MappingExample.labels
('Alice', 'Bob')
```

class enumagic.**StrEnum**

Bases: str, enum.Enum

Enum class that be used as a string.

Examples

```
>>> class StrExample(StrEnum):
...     A = 'Alice'
>>> StrExample.A.upper()
'ALICE'
```

__class__

alias of enum.EnumMeta

__str__() → str

Return the instance as a string.

Returns str – The value of the instance.

Examples

```
>>> str(StrExample.A)
'Alice'
```

CHAPTER 1

enumagic.django

Special enums for Django.

class enumagic.django.**ChoiceEnum**

Bases: str, enum.Enum

Enum class that can be used as Django field choices.

Examples

```
>>> from django.db.models import CharField, Model
>>> class ColorChoice(ChoiceEnum):
...     RED = '#F00'
...     GREEN = '#0F0'
...     BLUE = '#00F'
>>> class Color(Model):
...     color = CharField(choices=ColorChoice)
>>> example = Color(color=ColorChoice.RED)
>>> example.get_color_display()
'#F00'
```

do_not_call_in_templates = True

Prevent the Django template system from calling the enum.

__class__

alias of *enumagic.django.ChoiceMeta*

__str__() → str

Return the instance as a string.

Returns str – The name of the instance.

Examples

```
>>> str(ColorChoice.BLUE)
'BLUE'
```

__eq__(*other: Any*) → bool
Check whether the objects are equal.

Parameters **other** (Any) – Another object.

Returns bool – True if the objects are equal as strings, False otherwise.

Examples

```
>>> ColorChoice.GREEN == 'GREEN'
True
```

class enumagic.django.**ChoiceMeta**(**args*, ***kwargs*)
Bases: enum.EnumMeta

Choice enum metaclass.

__contains__(*item: Any*) → bool
Check whether the enum contains a certain item

Parameters **item** (Any) – A string or enum instance.

Returns bool – True if the enum has a member that matches the given item, False otherwise.

Raises TypeError – If the item is not a string or enum instance.

Examples

```
>>> 'GREEN' in ColorChoice
True
>>> 'CYAN' in ColorChoice
False
```

__getitem__(*name: str*) → str
Get the value of an enum member.

Parameters **name** (str) – The name of the item.

Returns str – The value of the instance.

Examples

```
>>> ColorChoice['GREEN']
'#0F0'
```

__iter__() → Iterator[Tuple[str, _VT]]
Iterate over the entries of the enum.

Yields tuple of str, object – The next tuple where the first element is the name of the enum instance and the second element is the value of the enum instance.

Examples

```
>>> it = iter(ColorChoice)
>>> next(it)
('RED', '#F00')
```

Python Module Index

e

Index

Symbols

C

D

E

I

L

M

S