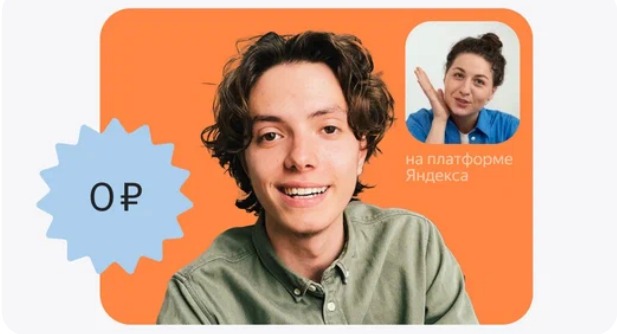


ХОЧУ ПОМОЧЬ  
ПРОЕКТУ

Что такое перечисления enum в Python



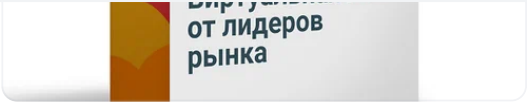
practicum.yandex.ru

РЕКЛАМА · 18+ Я

### Бесплатное занятие английским в Яндекс Практикуме


Полноценное занятие с преподавателем, а не презентация курсов

Узнать больше



900 ₽

Виртуальная АТС Базовая



2 000 ₽

Виртуальная АТС Расширенная

## Что такое перечисления enum в Python

о набор символических имен (членов), привязанных к уникальным постоянным значениям. сравниваться по идентичности, а само перечисление может повторяться.

бальные переменные, но предлагают более полезную функцию [repr\(\)](#), группировку, другие функции. Они наиболее полезны, когда есть переменная, которая может принимать ачений, например, дни недели или основные цвета RGB.

имволических имен (членов), привязанных к уникальным значениям, возвращать его члены в порядке определения, а (например Color(1)) для извлечения элементов по значению, а Color['RED'] для извлечения членов по имени.

с использованием [синтаксиса класса](#), либо с использованием [функционального синтаксиса](#):

```
from enum import Enum

# синтаксис класса
class Color(Enum):
    RED = 1
    GREEN = 2
    BLUE = 3

# функциональный синтаксис
Color = Enum('Color', ['RED', 'GREEN', 'BLUE'])
```

Несмотря на то, для создания перечислений может использоваться синтаксис класса, перечисления [не являются обычными классами Python](#).

Обратите внимание.

- Так как перечисления используются для представления констант, то рекомендуется использовать стиль написания имен для членов перечисления - [UPPER\\_CASE](#), так же этот стиль будет использоваться в примерах.

[Модуль enum](#) определяет пять классов перечислений, которые можно использовать для определения уникальных наборов имен и значений: enum.Enum, enum.IntEnum, enum.StrEnum, enum.Flag и enum.IntFlag. Он также определяет несколько декораторов и один помощник enum.auto.

**enum.Enum:**

Базовый класс enum.Enum предназначен для создания нумерованных констант.

**enum.IntEnum:**

Базовый класс enum.IntEnum предназначен для создания перечисляемых констант, которые являются подклассами [int](#).

**StrEnum:**

Базовый класс для создания перечислимых констант, которые также являются подклассами [str](#). (Добавлен в Python 3.11)

**enum.IntFlag:**

Базовый класс `enum.IntFlag` предназначен для создания перечисляемых констант, которые можно комбинировать с помощью побитовых операторов без потери членства в `enum.IntFlag`. Члены [базового класса `enum.IntFlag`](#) также являются подклассами [int](#).

РЕКЛАМА

mango-office.ru

MANGO OFFICE

MANGO OFFICE

Виртуальная АТС от лидеров рынка

900 ₹

Виртуальная АТС Базовая

MANGO OFFICE

MANGO OFFICE

Развивайте и масштабируйте бизнес легко

2 000 ₹

Виртуальная АТС Расширенная

enum имеет атрибуты, не конфликтующие с именами членов перечислений. (Добавлен в Python 3.11)

Обратите внимание, что атрибуты и члены перечисления должны быть определены в отдельных классах. Например, атрибуты `value` и `name` определены в классе `Enum`, а подклассы `Enum` могут определять элементы с именами `value` и `name`.

**@enum.verify():**  
[Декоратор `@enum.verify\(\)`](#) проверяет выбираемые пользователем ограничения перечисления. (Добавлен в Python 3.11)

Новое в Python 3.6: добавлены `enum.Flag`, `enum.IntFlag`, `enum.auto`

Новое в Python 3.11: добавлены `enum.StrEnum`, `enum.EnumCheck`, `enum.FlagBoundary`, `@enum.verify()`, `@enum.property()`.

## Заметки по `enum.IntEnum`, `enum.StrEnum` и `enum.IntFlag`

Эти три типа перечислений предназначены для замены существующих целочисленных и строковых значений. Как таковые, они имеют дополнительные ограничения:

- `__str__` использует значение, а не имя члена перечисления.
- `__format__` использует `__str__`, следовательно будет использовать значение члена перечисления вместо его имени

Если не нужны эти ограничения, то можно либо создать свой собственный базовый класс, смешав тип `int` или `str` самостоятельно:

```
from enum import Enum

class MyIntEnum(int, Enum):
    pass
```

или можно переназначить соответствующий `str()` и т. д. в своем перечислении:

```
from enum import IntEnum

class MyIntEnum(IntEnum):
    __str__ = IntEnum.__str__
```

Вверх

https://docs-python.ru/standart-library/modul-enum-perechislenija-python/

2/4

```

>>> from enum import Enum
>>> class Planet(Enum):
    REКЛАМА
    mango-office.ru
    2.4397e6)
    6.0518e6)
    6.37814e6)
    3.3972e6)
    7.1492e7)
    6.0268e7)
    2.5559e7)
    2.4746e7)
    s, radius):
        # in kilograms
    us # in meters

    lf):
        ational constant (m3 kg-1 s-2)

    ss / (self.radius * self.radius)

    ty

```

Содержание раздела:	
•	<a href="#">КРАТКИЙ ОБЗОР МАТЕРИАЛА.</a>
•	<a href="#">Создание перечислений enum.Enum</a>
•	<a href="#">Функциональный синтаксис определения перечислений</a>
•	<a href="#">Доступ к членам и их атрибутам перечисления модуля enum</a>
•	<a href="#">Дублирование членов и значений перечисления модуля enum</a>
•	<a href="#">Декоратор @enum.unique модуля enum</a>
•	<a href="#">Использование класса auto модуля enum</a>
•	<a href="#">Производные класса enum.Enum</a>
•	<a href="#">Итерация и сравнение перечислений модуля enum</a>
•	<a href="#">Допустимые члены и атрибуты enum</a>
•	<a href="#">Отличие enum от классов Python</a>
•	<a href="#">Примеры использования модуля enum</a>
•	<a href="#">Использование перечислений Python совместно с БД PostgreSQL</a>
•	<a href="#">Тонкости реализации модуля enum</a>
•	<a href="#">Вверх enumCheck модуля enum</a>
•	<a href="#">Класс FlagBoundary(.) модуля enum</a>

РЕКЛАМА

mango-office.ru

MANGO OFFICE

MANGO OFFICE

Виртуальная АТС от лидеров рынка

900 ₹

Виртуальная АТС Базовая

MANGO OFFICE

MANGO OFFICE

Развивайте и масштабируйте бизнес легко

2 000 ₹

Виртуальная АТС Расширенная

Вверх

https://docs-python.ru/standart-library/modul-enum-perechislenija-python/

4/4