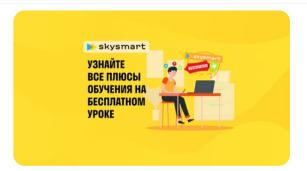
Сообщить об ошибке.

РЕКЛАМА

ХОЧУ ПОМОЧЬ ПРОЕКТУ

Модуль multipledispatch в Python



>> skysmart.ru

Программирование для детей и подростков в Skysmart

•

Узнать больше

.....

30-dlyaVAS

Модуль multipledispatch в Python

C₩ cutwoodshop.ru

промокод!

3D карта мира из дерева по специальной цене!

%

Отличный подарок вам и вашим близким. Заказывайте на сайте используя промокод!

Подсветка

Ручная работа Гарантия качества

Купить ен свку.

регрузка методов/функций в Python

г шаблона программирования множественной диспетчеризации (перегрузки методов и функций) в из во избежание конфликтов и обеспечивает дополнительную поддержку пространства имен.

относящиеся к ключевым словам.

<u>ра</u>.

й, например (int, float).

рактные классы, например, Iterator, Number,...

о повторного поиска.

начности во время выбора реализации метода/функции.

нения двусмысленностей, когда они возникают.

ен с необязательными аргументами ключевых слов.

Установка модуля multipledispatch в виртуальное окружение:

```
# создаем виртуальное окружение, если нет
$ python3 -m venv .venv --prompt VirtualEnv
# активируем виртуальное окружение
$ source .venv/bin/activate
# ставим модуль multipledispatch
(VirtualEnv):~$ python3 -m pip install -U multipledispatch
```

Содержание:

- <u>Пример использования модуля multipledispatch</u>.
- Разрешение перегруженных методов/функций.
 - Реализация метода/функции для нескольких типов.
 - Использование абстрактных типов.
 - Выбор конкретной реализации.
- Множественная отправка.
 - Множественная вариативная отправка.
 - Неоднозначность выбора реализации перегруженной функции.

Пример использования модуля multipledispatch.

Пример перегрузки функций:

```
from multipledispatch import dispatch

@dispatch(object, object)

# базовая реализация функции `add()`

def add(x, y):
    return f"{x} + {y}"

@dispatch(int, int)

# реализация функции `add()` для целых чисел

def_add(x, y):
    BBepx n x + y
```

```
>>> add(1, 2)
# 3
>>> add(1, 'hello')
                                patch
                30-dlyaVAS
                                 `.add()`
  промокод!
 C₩ cutwoodshop.ru
 3D карта мира из дерева
 по специальной цене!
                                 для целых чисел
 %
 Отличный подарок вам и вашим
 близким. Заказывайте на сайте
 используя промокод!
 Ручная работа
 Гарантия качества
 Подсветка
                               енных методов/функций.
```

...ожественняя длене периводиния правка выбирает метод/функцию, на основе анализа типов входящих аргументов.

```
from multipledispatch import dispatch

@dispatch(int)
# реализация функции `f()` для целого числа

def f(x):
# увеличит целое число
return x + 1

@dispatch(float)
# реализация функции `f()` для вещественного числа

def f(x):
# уменьшит вещественное число
return x - 1

>>> f(1)
# 2

>>> f(1.0)
# 0.0
```

Реализация метода/функции для нескольких типов.

Аналогично встроенной <u>функции isinstance()</u>, в декораторе @dispatch() указываются несколько допустимых типов с помощью кортежа. Приведенная ниже реализация f() для (list, tuple) применит реализацию f() для целого числа к каждому элементу в списке или кортеже, переданному в качестве аргумента.

```
@dispatch((list, tuple))
# реализация функции `f()` для типов `list` и `tuple`

def f(x):
    """ Применит `f(y: int)` к каждому элементу в списке или кортеже """
    return [f(y) for y in x]

>>> f([1, 2, 3])
# [2, 3, 4]
>>> f((1, 2, 3))
# [2, 3, 4]
```

Вверх

Использование абстрактных типов.

В декораторе @dispatch() можно также использовать абстрактные классы, такие как Iterable и Number, вместо <u>типов</u> объединения, таких как (list, tuple) или (int, float) соответственно.



C₩ cutwoodshop.ru

3D карта мира из дерева по специальной цене!

имых реализаций, то используется наиболее конкретная. В следующем примере создается ных итераций.

% patch able Отличный подарок вам и вашим близким. Заказывайте на сайте используя промокод! Ручная работа Гарантия качества Подсветка Купить

in L], [])

зации.

```
>>> flatten([1, [2], 3])
# [1, 2, 3]
>>> flatten([1, 2, (3, 4), [[5]], [(6, 7), (8, 9)]])
# [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Так как строки итерируемы, то они тоже будут "сглажены":

```
>>> flatten([1, 'hello', 3])
# [1, 'h', 'e', 'l', 'l', 'o', 3]
```

Нужно избегать этого, конкретизируя реализацию flatten() до типа str. Поскольку тип str более конкретен, чем Iterable, то следующая реализация имеет приоритет для типа str.

```
@dispatch(str)
def flatten(s):
    return s
>>> flatten([1, 'hello', 3])
# [1, 'hello', 3]
```

Модуль multipledispatch зависит от механизма работы функции issubclass(), который определяет, какие типы более специфичны, чем другие.

Множественная отправка.

Все эти правила применяются, когда метод/функция принимает несколько аргументов в качестве входных данных.

```
from multipledispatch import dispatch
@dispatch(object, object)
def f(x, y):
    return x + y
@dispatch(object, float)
def f(x, y):
    """ Квадрат второго аргумента, если это `float` """
    return x + γ**2
   Вверх
>>> f(1, 10)
```

```
12.09.2023, 15:38

# 11

>>> f(1.0, 10.0)

# 101.0
```

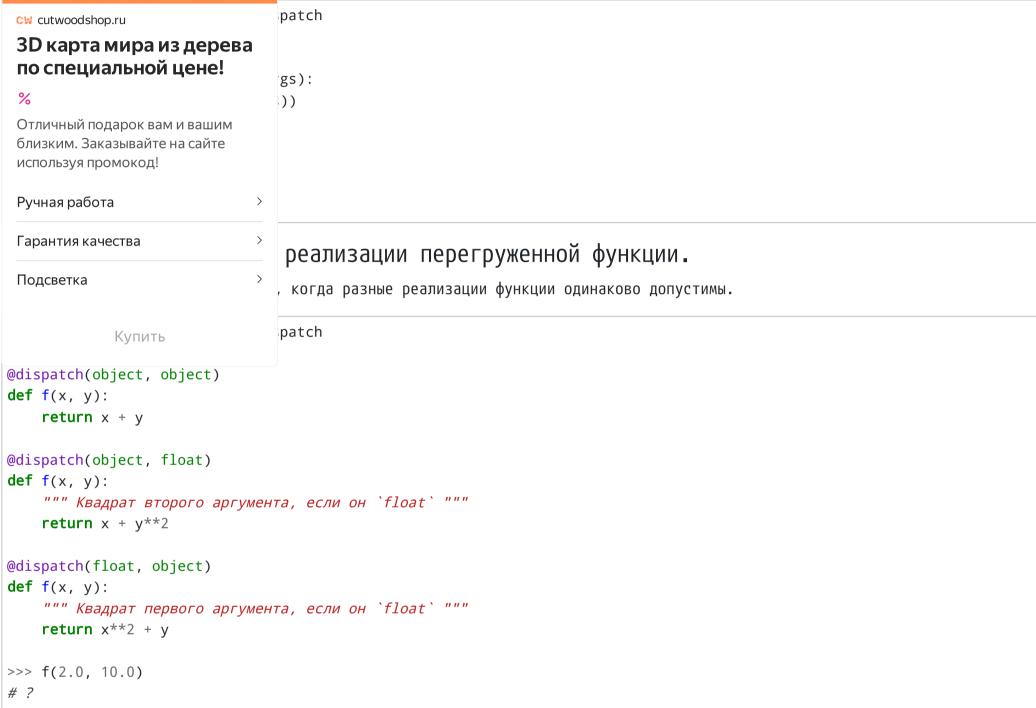


ная отправка.

зает множественную отправку (включая поддержку типов объединения) в качестве последнего

цноэлементным списком, содержащим тип аргументов, которые принимает функция.

г число с плавающей запятой, за которым следует любое число (включая 0) либо int, либо



Какого результата ждать: 2,0 ** 2 + 10,0 или 2,0 + 10,0 ** 2? Типы входных данных удовлетворяют трем различным реализациям, две из которых имеют одинаковую силу.

```
input types: float, float
Вариант 1: object, object
Вариант 2: object, float
Вариант 3: float, object
```

Вариант 1 - строго менее конкретен, чем варианты 2 или 3, поэтому он отбрасывается. Но варианты 2 и 3 одинаково специфичны, поэтому неясно, какой из них использовать.

Чтобы решить такие проблемы, как эта, множественная отправка проверяет предоставленные ей сигнатуры типов и ищет неоднозначности. Затем появляется предупреждение, подобное следующему:

```
multipledispatch/dispatcher.py:27: AmbiguityWarning:
Ambiguities exist in dispatched function f

The following signatures may result in ambiguous behavior:
        [object, float], [float, object]

Cd BBepx making the following additions:
```

@dispatch(float, float)
def f(...)
 warn(warning_text(dispatcher.name, ambiguities), AmbiguityWarning)
14.0



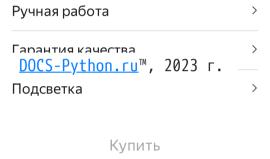
огда в перегруженном методе/функции есть двусмысленность, а так же помогает создать /чае функция с сигнатурой (float, float) более конкретна, чем варианты 2 или 3, и таким збежать этого предупреждения, необходимо создать эту реализацию раньше других.

C₩ cutwoodshop.ru

3D карта мира из дерева по специальной цене!



Отличный подарок вам и вашим близким. Заказывайте на сайте используя промокод!



, то одна из конкурирующих функций будет выбрана псевдослучайно. По умолчанию, выбор согласован во время сеанса интерпретатора, но может меняться от сеанса к сеансу.

(Внимание! При копировании материала ссылка на источник обязательна)

@docs_python_ru