Разработка программного обеспечения на языке Python

Библиотеки, репозитории, инструменты разработчика

Библиотеки в Python

00:00 / 03:55

PyPI

С Python Package Index (PyPI) - каталогом пакетов Python вы можете познакомиться по адресу

Давайте рассмотрим несколько библиотек из числа наиболее часто скачиваемых библиотек из этого каталога и посмотрим, что они могут делать.

№ п/п	Наименование библиотеки
1	urllib3
2	requests
3	python-dateutil
4	idna
5	pip

6	numpy
7	cryptography
8	pandas
9	scipy
10	more-itertools

urllib3

Наиболее популярная библиотека для работы с запросами.

Позволяет как получать информацию по запросу

```
>>> import urllib3
```

Создаем менеджер пула запросов

```
>>> http = urllib3.PoolManager()
```

Создаем `"get"` запрос для получения данных с сайта

```
>>> r = http.request('GET', 'http://httpbin.org/robots.txt')
>>> r.data
```

В результате получим:

```
b'User-agent: *\nDisallow: /deny\n'
```

Также можно выполнять запросы для отправки информации на сайт. Например, заполнения формы, или авторизации:

```
>>> r = http.request('POST', 'http://httpbin.org/post', fields={'hello': 'world'})
```

Библиотека поддерживает еще множество возможностей и настроек: извлечение данных из `url`, получение данных в разных форматах, обработка ошибок и т.п. Об остальных возможностях вы можете почитать в документации

requests

Это также библиотека для работы с запросами. Имеет несколько меньший спектр функциональности, чем `urllib3`, но является более простой в освоении и использовании. Какую из этих двух библиотек применять можно решить лишь поработав с обоими.

```
>>> import requests
>>> r = requests.get('https://api.github.com/user', auth=('user', 'pass'))
>>> r.status_code
200
>>> r.headers['content-type']
'application/json; charset=utf8'
>>> r.encoding
'utf-8'
>>> r.text
'{"type":"User"...'
>>> r.json()
{'disk_usage': 368627, 'private_gists': 484, ...}
```

Подробнее с возможностями библиотеки можно ознакомится в документации. Нужно отметить, что она настолько популярна в русскоязычном сегменте, что сообщество разработчиков создало документацию на русском языке.

python-dateutil

Библиотека предоставляет мощные расширения для стандартного модуля `datetime`, доступного в Python. Реализован широкий функционал работы с датами, временем, часовыми поясами, временными дельтами.

```
>>> from dateutil.parser import *
>>> now = parse("Sat Oct 11 17:13:46 UTC 2003")
>>> today = now.date()
>>> print(f"Сегодня:{today}")
Сегодня: 2003-10-11
```

С полным функционалом библиотеки можно познакомится в документации

idna

Поддерживает протокол интернационализированных доменных имен в приложениях. Обеспечивает корректное отображение доменных имен сайтов в латиницу и обратно.

```
>>> import idna
>>> idna.encode('ドメイン.テスト')
b'xn--eckwd4c7c.xn--zckzah'
>>> print(idna.decode('xn--eckwd4c7c.xn--zckzah'))
ドメイン.テスト
```

Подробнее познакомиться с библиотекой можно в документации

pip

Несмотря на то что эта библиотека не первая по числу скачиваний, ее можно назвать самой популярной. Именно эта библиотека дает возможность устанавливать любые другие библиотеки.

Поэтому если все остальные библиотеки устанавливаются с помощью команды:

```
pip install имя библиотеки
```

То поставить `pip` можно командой:

```
python -m pip –version
```

С помощью `pip` можно поставить не только последнюю версию библиотеки, но и какую-то конкретную:

```
pip install SomePackage==1.0.4
```

Или установить все библиотеки из некоего списка зависимостей, хранящегося в файле:

```
pip install -r requirements.txt
```

Подробнее в документации

numpy

Мощная оптимизированная библиотека для работы с -мерными массивами. Обеспечивает быстрые и высокопроизводительные математические операции из раздела линейной алгебры и тензорных вычислений, а также операции применяемые к массивам поэлементно. Позволяет работать со случайными числами и многое другое.

```
>>> import numpy as np
```

Создадим вектор со значениями от до и преобразуем его в матрицу.

```
>>> a = np.arange(15).reshape(3, 5)
>>> a
array([[ 0,  1,  2,  3,  4],
          [ 5,  6,  7,  8,  9],
         [10, 11, 12, 13, 14]])
>>> a.shape
(3, 5)
>>> a.ndim
2
```

Применим функцию к массиву:

```
>>> B = np.arange(3)
>>> B
array([0, 1, 2])
>>> np.exp(B)
array([1.        , 2.71828183, 7.3890561 ])
```

Подробнее о библиотеке

cryptography

Данная библиотека включает множество алгоритмов для шифрования данных, а также позволяет работать с низкоуровневыми операциями шифрования, для создания своих алгоритмов.

```
>>> from cryptography.fernet import Fernet
>>> key = Fernet.generate_key()
>>> f = Fernet(key)
>>> token = f.encrypt(b"A really secret message. Not for prying eyes.")
>>> token
b'gAAAAABfzQp02IVJ4OWEZ6-TY7XxB_pj3P5olJT2lQ7ZKcNXSe5BwTBbAy-
EDyJCsGKP6W0xkBOoXKUbXAJygAUaT3AVfaCEveWmIhTHC_-ZMu3Ux0C5MED84dFAXQsZjh8qrZg4B2RS'
>>> f.decrypt(token)
b'A really secret message. Not for prying eyes.'
```

Подробнее в документации

pandas

Дает быстрые, гибкие и очень удобные инструменты для работы со структурированными данными (табличными, многомерными и потенциально неоднородными). Наиболее часто применяемая библиотека для анализа данных, предварительной обработки данных. Поддерживает чтение и запись множества форматов хранения данных. Вычисление массивов содержащих однотипные данные опирается на вычислительное ядро библиотеки `numpy`.

```
>>> import numpy as np
>>> import pandas as pd
```

Создадим ключевой элемент работы с данными в `pandas` - DataFrame. Для создания набора данных используем генератор случайных массивов библиотеки `numpy`:

```
>>> df = pd.DataFrame(np.random.randn(6, 4), index=dates, columns=list('ABCD'))
>>> df
          A         B         C         D
0.469112 -0.282863 -1.509059 -1.135632
1.212112 -0.173215  0.119209 -1.044236
-0.861849 -2.104569 -0.494929  1.071804
0.721555 -0.706771 -1.039575  0.271860
-0.424972  0.567020  0.276232 -1.087401
-0.673690  0.113648 -1.478427  0.524988
```

Посмотрим базовые статистики описывающие полученный набор данных:

```
>>> df.describe()
              A         B         C         D
count  6.000000  6.000000  6.000000  6.000000
mean   0.073711 -0.431125 -0.687758 -0.233103
std    0.843157  0.922818  0.779887  0.973118
min   -0.861849 -2.104569 -1.509059 -1.135632
25%   -0.611510 -0.600794 -1.368714 -1.076610
50%    0.022070 -0.228039 -0.767252 -0.386188
75%    0.658444  0.041933 -0.034326  0.461706
max    1.212112  0.567020  0.276232  1.071804
```

Подробнее о библиотеке

scipy

Библиотека для вычислений в сфере математики, естественных наук и инженерии. Также как и `pandas` она зависит от `numpy`. Но дополнительно имеет гораздо более богатый инструментарий для статистических вычислений, численного интегрирования, оптимизации и работы с разреженными данными. Со связкой библиотек SciPy и NumPy работают ведущие математики и инженеры мира.

Вычислим косинусное расстояние между двумя векторами.

```
>>> from scipy.spatial import distance
>>> distance.cosine([1, 0, 0], [0, 1, 0])
1.0
>>> distance.cosine([1, 0, 0], [1, 0, 0])
0.0
```

Подробнее о библиотеке

more-itertools

Расширяет возможности и без того очень полезной встроенной библиотеки itertools. В ней есть дополнительные функции, процедуры и строительные блоки.

Разделим последовательность на N частей:

```
>>> from more_itertools import chunked
>>> iterable = [0, 1, 2, 3, 4, 5, 6, 7, 8]
>>> list(chunked(iterable, 3))
[[0, 1, 2], [3, 4, 5], [6, 7, 8]]
```

Подробнее о библиотеке

Приведенный список библиотек не является последовательным списком из наиболее часто загружаемых библиотек. Применение некоторых библиотек тяжело показать в рамках такого короткого текста, а некоторые, не смотря на свое частое скачивание, имеют достаточно узкое применение. Можете познакомится с более подробным списком

Создание своей библиотеки

ПРЕДЫДУЩИЙ ЭЛЕМЕНТ КУРСА

◄ Задание 13. Властелин колец

Перейти на...

СЛЕДУЮЩИЙ ЭЛЕМЕНТ КУРСА

Вопросы к лекции Библиотеки, репозитории, инструменты разработчика ►