Сообщить об ошибке.

РЕКЛАМА • 16+

хочу помочь

Mogyntroselectors в Python, готовность файлов к вводу/выводу



S skillbox.ru

Обучение Data Scientist - Обновили курс в 2023 году!

Станьте дата-инженером, аналитиком данных или ML-инженером. з/п от 120 000 руб.

Узнать больше



/ Модуль selectors в Python, готовность файлов к вводу/выводу

ости ввода/вывода нескольких файловых объектов

высокоуровневое и эффективное мультиплексирование ввода-вывода, основанное на и пользователям не нужно точный контроль над примитивами операционной системы, то от модуль вместо модуля select.

есколько конкретных реализаций: <u>selectors.EpollSelector()</u>, <u>selectors.SelectSelector()</u> зовать для ожидания уведомления о готовности ввода-вывода для нескольких файловых <u>ект</u>" относится к любому объекту с методом <u>file.fileno()</u> или необработанным файловым

or - это псевдоним наиболее эффективной реализации, доступной на текущей платформе: м по умолчанию для большинства пользователей.

12+ x файловых объектов зависит от платформы: в Windows поддерживаются работа только с сокетами, но не с каналами, тогда как в Unix поддерживаются оба варианта. Также могут поддерживаться некоторые другие типы, например FIFO или специальные файловые устройства.

События представляют собой побитовую маску, указывающую, какие события ввода-вывода следует ожидать для данного файлового объекта. Это может быть комбинация констант модуля:

- selectors.EVENT_READ доступно для чтения;
- selectors.EVENT_WRITE доступно для записи;

Классы определяемые модулем.

<u>Иерархия классов модуля</u>:

BaseSelector

Пρ

- +-- SelectSelector
- +-- PollSelector
- +-- EpollSelector
- +-- DevpollSelector
- +-- KqueueSelector

Абстрактный базовый класс <u>selectors.BaseSelector</u> и все его конкретные реализации поддерживают <u>протокол диспетчера</u> контекста.

selectors.SelectorKey():

Класс selectors. SelectorKey() - это именованное множество, используемое для связывания файлового объекта с его базовым файловым дескриптором, выбранной маской события и вложенными данными. Он возвращается несколькими методами BaseSelector.

<u>Класс selectors.SelectorKey() следующие атрибуты</u>:

- SelectorKey.fileobj зарегистрированный файловый объект.
- SelectorKey.fd базовый файловый дескриптор.

ntorKey.events - события, которые необходимо дождаться для этого файлового объекта.

• SelectorKey.data - необязательные непрозрачные данные, связанные с этим файловым объектом: например, они могут использоваться для хранения идентификатора сеанса для каждого клиента.

selectors.DefaultSelector():

Knacc selectors.DefaultSelector(), автоматически выбирает и использует наиболее эффективную реализацию, доступную на текущей платформе. Этот класс должен быть выбором по умолчанию для большинства пользователей.

Наследует методы от абстрактного класса |selectors.BaseSelector()|.

selectors.SelectSelector():

Класс selectors.DefaultSelector() создан на основе базового объекта <u>select.select()</u>.

Наследует методы от абстрактного базового класса selectors. BaseSelector.

selectors.PollSelector():

Класс selectors.PollSelector() создан на основе базового объекта <u>select.poll()</u>.

Наследует методы от абстрактного базового класса selectors. BaseSelector.

selectors.EpollSelector():

Класс selectors.EpollSelector() создан на основе базового объекта <u>select.epoll()</u>.

Наследует методы от абстрактного базового класса <u>selectors.BaseSelector</u> и определяет один дополнительный метод:

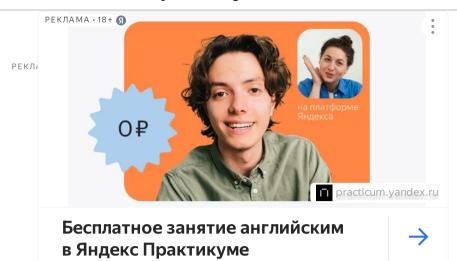
• EpollSelector.fileno() - возвращает файловый дескриптор, используемый базовым объектом select.epoll().

Пример неблокирующего сокет сервера с использованием модуля selectors.

Вот простой эхо сервер.

```
import selectors
import socket
sel = selectors.DefaultSelector()
def accept(sock, mask):
    # Должен быть готов к чтению
    conn, addr = sock.accept()
   print('Coздан coкeт:', conn, 'для адреса:', addr)
    # включаем для него неблокирующий режим
    conn.setblocking(False)
    # регистрируем для прослушивания событий клиента
    sel.register(conn, selectors.EVENT_READ, read)
def read(conn, mask):
    # Должен быть готов к чтению
    data = conn.recv(1000)
    if data:
        print('Отвечаем: ', repr(data), 'сокету =>', conn)
        conn.send(data)
    else:
        print('Закрытие соединения', conn)
        sel.unregister(conn)
        conn.close()
sock = socket.socket()
sock.bind(('localhost', 8008))
sock.listen(5)
sock.setblocking(False)
sel.register(sock, selectors.EVENT_READ, accept)
while True:
   events = sel.select()
   BBepx ey, mask in events:
```

callback = key.data
callback(key.fileobj, mask)





Kypc: Data Scientist - скидки до 40%!



Содержание раздела:

- <u>КРАТКИЙ ОБЗОР МАТЕРИАЛА.</u>
- <u>Класс BaseSelector() модуля selectors</u>

<u>DOCS-Python.ru</u>™, 2023 г.

(Внимание! При копировании материала ссылка на источник обязательна)

@docs_python_ru

