信号

Scrapy广泛使用信号来通知特定事件发生的时间。您可以在Scrapy项目中捕获一些这些信号（例如使用扩展名）来执行其他任务或扩展Scrapy以添加未提供的开箱即用的功能。

即使信号提供了几个参数，捕获它们的处理程序也不需要接受所有这些参数 - 信号调度机制只会传递处理程序接收的参数。

您可以通过Signals API连接信号（或发送您自己的 信号）。

这是一个简单的示例，展示了如何捕获信号并执行某些操作：

```
from scrapy import signals
from scrapy import Spider


class DmozSpider(Spider):
    name = "dmoz"
    allowed_domains = ["dmoz.org"]
    start_urls = [
        "http://www.dmoz.org/Computers/Programming/Languages/Python/Books/",
        "http://www.dmoz.org/Computers/Programming/Languages/Python/Resources/",
    ]


    @classmethod
    def from_crawler(cls, crawler, *args, **kwargs):
        spider = super(DmozSpider, cls).from_crawler(crawler, *args, **kwargs)
        crawler.signals.connect(spider.spider_closed, signal=signals.spider_closed)
        return spider


    def spider_closed(self, spider):
        spider.logger.info('Spider closed: %s', spider.name)


    def parse(self, response):
        pass
```

延迟信号处理程序

有些信号支持从处理程序返回Twisted延迟，请参阅下面的内置信号参考以了解哪些信号。

内置信号参考

这是Scrapy内置信号列表及其含义。

engine_started

scrapy.signals.engine_started ( )

Scrapy引擎开始爬网时发送。

此信号支持从其处理程序返回延迟。

注意

这个信号可能被解雇*后*的 `spider_opened` 信号，这取决于蜘蛛是如何开始。因此，**不要**依赖此信号之前被解雇 `spider_opened` 。

engine_stopped

scrapy.signals.engine_stopped ( )

Scrapy引擎停止时发送（例如，爬网过程完成时）。

此信号支持从其处理程序返回延迟。

item_scraped

scrapy.signals.item_scraped (*项目* , *响应* , *蜘蛛*)

在物品已经通过所有物品管道阶段（未被丢弃）之后，在物品被刮除时发送 。

此信号支持从其处理程序返回延迟。

参数：

- item（字典或 `Item` 对象） - 刮下的项目
- **蜘蛛**（ `Spider` 物体） - 刮掉物品的蜘蛛
- response（ `Response` object） - 项目被刮取的响应

item_dropped

scrapy.signals.item_dropped (*项目* , *响应* , *异常* , *蜘蛛*)

在某个阶段引发异常时，从项目管道中删除项目后发送 `DropItem` 。

此信号支持从其处理程序返回延迟。

参数：

- item（字典或 `Item` 对象） - 从项目管道中删除的项目
- **蜘蛛**（ `Spider` 物体） - 刮掉物品的蜘蛛
- response（ `Response` object） - 项目被删除的响应
- exception（ `DropItem` exception） - `DropItem` 导致项被删除的异常（必须是 子类）

spider_closed

scrapy.signals.spider_closed(*蜘蛛，原因*)

蜘蛛关闭后发送。这可用于释放保留的每蜘蛛资源`spider_opened`。

此信号支持从其处理程序返回延迟。

参数：
- **蜘蛛**（`Spider`对象） - 已经关闭的蜘蛛
- **reason**（*str*） - 描述蜘蛛关闭原因的字符串。如果由于蜘蛛完成刮擦而关闭，原因是`'finished'`。否则，如果通过调用`close_spider`引擎方法手动关闭了spider，那么原因是在该`reason`方法的参数中传递的那个（默认为`'cancelled'`）。如果引擎关闭（例如，通过按Ctrl-C来停止它）原因将是`'shutdown'`。

spider_opened

scrapy.signals.spider_opened(*蜘蛛*)

在蜘蛛打开爬行后发送。这通常用于保留每个蜘蛛资源，但可用于打开蜘蛛时需要执行的任何任务。

此信号支持从其处理程序返回延迟。

参数： **蜘蛛**（`Spider`对象） - 已经打开的蜘蛛

spider_idle

scrapy.signals.spider_idle(*蜘蛛*)

当蜘蛛闲置时发送，这意味着蜘蛛没有进一步：

- 请求等待下载
- 请求预定
- 项目管道中正在处理的项目

如果在此信号的所有处理程序完成后空闲状态仍然存在，则引擎开始关闭蜘蛛。蜘蛛完成关闭后，`spider_closed`信号被发送。

您可以引发`DontCloseSpider`异常以防止蜘蛛被关闭。

此信号不支持从其处理程序返回延迟。

参数： **蜘蛛**（`Spider`对象） - 已经空闲的蜘蛛

注意

在`spider_idle`处理程序中调度某些请求并 **不能**保证它可以阻止蜘蛛被关闭，尽管它有时可以。这是因为如果调度程序拒绝所有调度的请求（例如，由于重复而过滤），则蜘蛛可能仍然保持空闲。

spider_error

scrapy.signals.spider_error (*失败*, *回应*, *蜘蛛*)

蜘蛛回调生成错误时发送（即引发异常）。

此信号不支持从其处理程序返回延迟。

参数：
- **failure**（Failure对象）- 作为Twisted Failure对象引发的异常
- **response**（`Response` object）- 引发异常时正在处理的响应
- **spider**（`Spider` object）- 引发异常的蜘蛛

request_scheduled

scrapy.signals.request_scheduled (*请求*, *蜘蛛*)

在引擎安排时发送`Request`，稍后再下载。

该信号不支持从其处理程序返回延迟。

参数：
- **request**（`Request` object）- 到达调度程序的请求
- **spider**（`Spider` object）- 产生请求的蜘蛛

request_dropped

scrapy.signals.request_dropped (*请求*, *蜘蛛*)

当`Request`稍后要下载的引擎调度的a 被调度程序拒绝时发送。

该信号不支持从其处理程序返回延迟。

参数：
- **request**（`Request` object）- 到达调度程序的请求
- **spider**（`Spider` object）- 产生请求的蜘蛛

response_received

scrapy.signals.response_received (*响应*, *请求*, *蜘蛛*)

当引擎`Response`从下载程序收到新内容时发送。

此信号不支持从其处理程序返回延迟。

参数：

- **响应**（`Response`对象）- 收到的响应
- **request**（`Request` object）- 生成响应的请求
- **spider**（`Spider` object）- 响应所针对的蜘蛛

response_downloaded

`scrapy.signals.response_downloaded`（*响应，请求，蜘蛛*）

下载后立即由下载程序发送`HTTPResponse`。

此信号不支持从其处理程序返回延迟。

参数：

- **响应**（`Response`对象）- 下载的响应
- **request**（`Request` object）- 生成响应的请求
- **spider**（`Spider` object）- 响应所针对的蜘蛛