统计收集

Scrapy提供了一种方便的工具，用于以键/值的形式收集统计数据，其中值通常是计数器。该工具称为统计收集器，可以通过Crawler API的 `stats` 属性进行访问，如下面的Common Stats Collector使用部分中的示例所示。

但是，统计信息收集器始终可用，因此无论是否启用统计信息收集，您始终可以将其导入模块并使用其API（增加或设置新的统计信息键）。如果它被禁用，API仍然可以工作，但它不会收集任何东西。这旨在简化统计信息收集器的使用：您应该花费不超过一行代码来收集蜘蛛，Scrapy扩展或您使用统计信息收集器的任何代码中的统计信息。

统计信息收集器的另一个特性是它非常有效（启用时）并且在禁用时非常有效（几乎不可察觉）。

统计收集器为每个打开的蜘蛛保留一个统计表，当蜘蛛打开时它会自动打开，当蜘蛛关闭时它会关闭。

Common Stats Collector使用

通过 `stats` 属性访问统计信息收集器。以下是访问统计信息的扩展示例：

```
class ExtensionThatAccessStats(object):

    def __init__(self, stats):
        self.stats = stats

    @classmethod
    def from_crawler(cls, crawler):
        return cls(crawler.stats)
```

设置统计值：

```
stats.set_value('hostname', socket.gethostname())
```

增量统计值：

```
stats.inc_value('custom_count')
```

仅在大于上一个时设置统计值：

```
stats.max_value('max_items_scraped', value)
```

仅在低于上一个值时设置统计值：

```
stats.min_value('min_free_memory_percent', value)
```

获取统计值：

```
>>> stats.get_value('custom_count')
1
```

获取所有统计数据：

```
>>> stats.get_stats()
{'custom_count': 1, 'start_time': datetime.datetime(2009, 7, 14, 21, 47, 28, 977139)}
```

可用的统计数据收集器

除了基本功能外`StatsCollector`，Scrapy还提供其他统计数据收集器，可扩展基本的统计数据收集器。您可以通过该`STATS_CLASS`设置选择要使用的统计信息收集器。使用的默认统计信息收集器是`MemoryStatsCollector`。

MemoryStatsCollector

类 **scrapy.statscollectors.MemoryStatsCollector**

一个简单的统计数据收集器，用于在关闭后保留内存中最后一次抓取（每个蜘蛛）的统计数据。可以通过`spider_stats`属性访问统计信息，该属性是由蜘蛛域名键入的dict。

这是Scrapy中使用的默认统计信息收集器。

spider_stats

dicts的一个词典（由蜘蛛名称键入），包含每个蜘蛛的最后一次刮擦运行的统计数据。

DummyStatsCollector

类 **scrapy.statscollectors.DummyStatsCollector**

一个Stats收集器，除了非常有效之外什么都不做（因为它什么都不做）。可以通过设置设置此统计信息收集器`STATS_CLASS`，以禁用统计信息收集以提高性能。但是，与其他Scrapy工作负载（如解析页面）相比，统计信息收集的性能损失通常很小。