CRAWLING THE CONCRETE JUNGLE

WITH SCRAPY

OVERVIEW

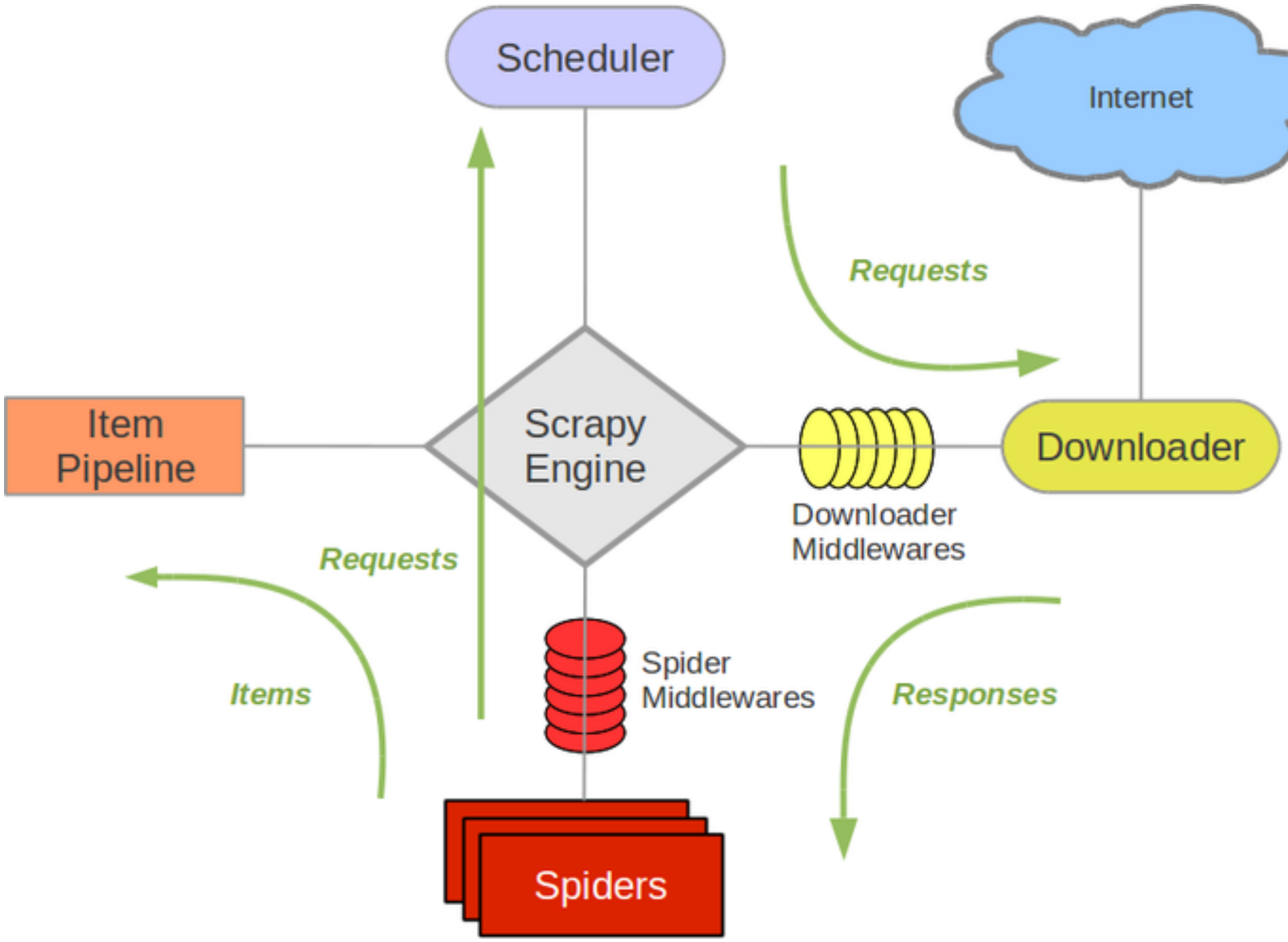
StreetEasy

- Aggregates NYC listing data
- August 2013 - Purchased by Zillow for $50mm
- Standardized listing data
- Over 19k active rental listings

PROJECT OVERVIEW

- Used Scrapy to crawl and parse 19k active rental listings
- Scrapy shell & Chrome
- The dataLayer
- Pipe data to PostgreSQL with SQLAlchemy
- Limitations / Future Improvements
  - Only rental data / expand to sale data
  - Current snapshot / scrape historical listing data
  - Machine learning to predict NYC real estate market

SCRAPY ARCHITECTURE



SPIDER CODE SNIPPET

```
import scrapy
import re
import ast
from nycrex.items import listingItem

import logging
from scrapy.utils.log import configure_logging

configure_logging(install_root_handler=False)
logging.basicConfig(
    filename='nycrex.log',
    format='%(asctime)s: %(name)s: %(levelname)s: %(message)s',
    level=logging.INFO
)
logging.getLogger('sqlalchemy.engine').setLevel(logging.DEBUG)

class StreeteasySpider(scrapy.Spider):
    name = "streeteasy"
    allowed_domains = ["streeteasy.com"]
    start_urls = [
        'http://streeteasy.com/for-rent/manhattan',
        'http://streeteasy.com/for-rent/brooklyn',
        'http://streeteasy.com/for-rent/queens',
        'http://streeteasy.com/for-rent/bronx',
        'http://streeteasy.com/for-rent/staten-island'
        ]

    def parse(self, response):
        for sel in response.xpath('//div[@class="details-title"]'):

            url = 'http://streeteasy.com%s' %sel.xpath('a/@href').extract()[0]
            yield scrapy.Request(url, callback=self.parseListing)

        for href in response.xpath('//div[@class="pagination center_aligned bot
            url = response.urljoin(href.extract())
            yield scrapy.Request(url, callback=self.parse)


    def parseListing(self, response):

        # Parse <script> dataLayer object
        data = response.xpath('//script/text()[contains(., "dataLayer")]')[0]
        data = data.extract().strip()
```

SHINY DEMO