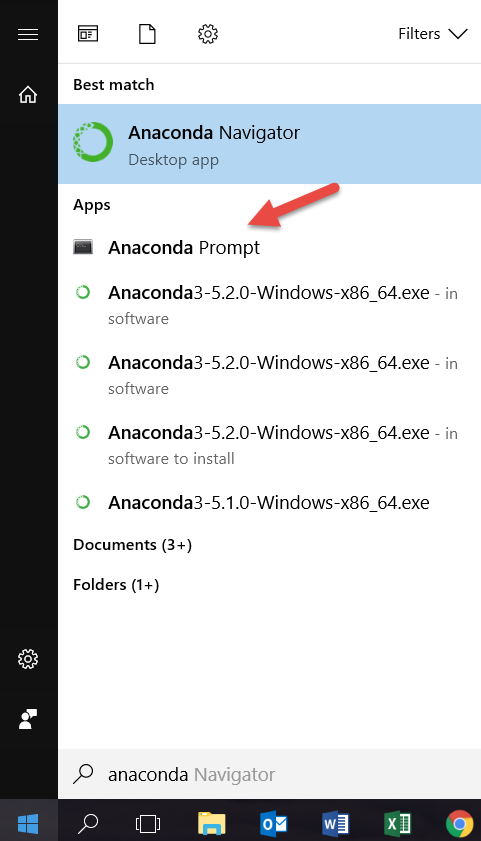
**Lab: Web crawling and scraping with Scrapy**

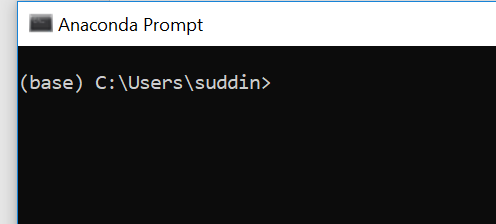
This Lab will take you through the following steps:

1. Installing Scrappy on Windows
2. Creating a new Scrapy project
3. Writing a spider to crawl a site and extract data
4. Exporting the scraped data using the command line
5. Changing spider to recursively follow links
6. **Installing Scrappy on Windows**

If you want to download Scrapy on your windows machine, you can install it using Anaconda as described below:

Click on search C:\Users\MHUMAY~1\AppData\Local\Temp\SNAGHTML858b04f.PNG and start typing “**anaconda prompt”:**





Now type the following in ‘Anaconda Prompt’:

> conda install -c anaconda scrapy

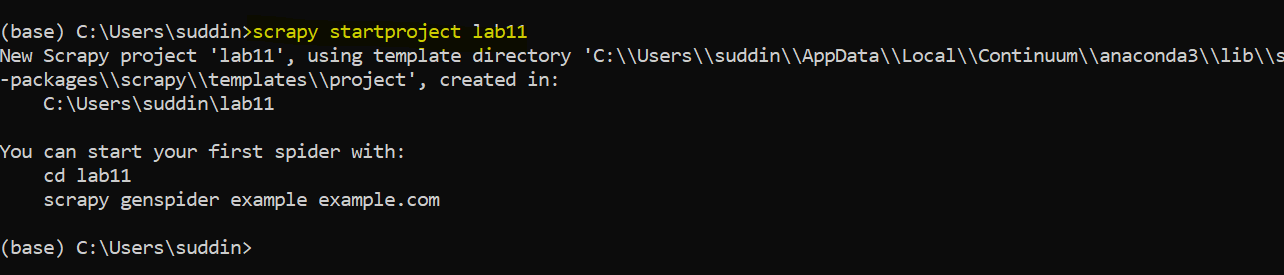
Press Y when prompted.

1. **Creating a new Scrapy project**

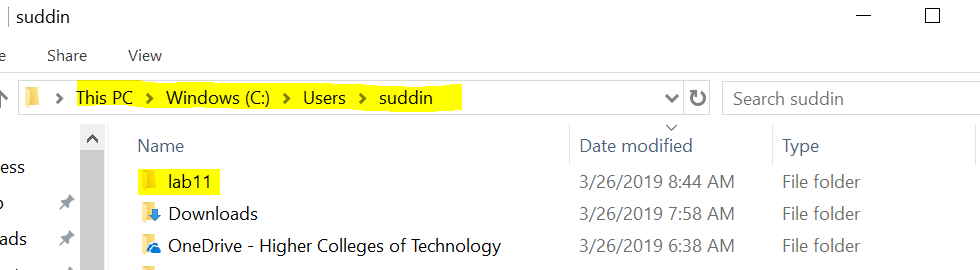
In order to set up a new Scrapy project. Enter a directory where you’d like to store your code and run.

> scrapy startproject lab11

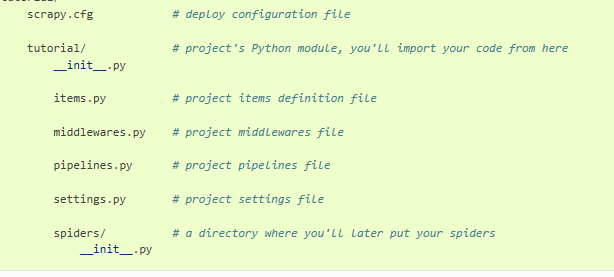
*Where lab11 is the name of the project*

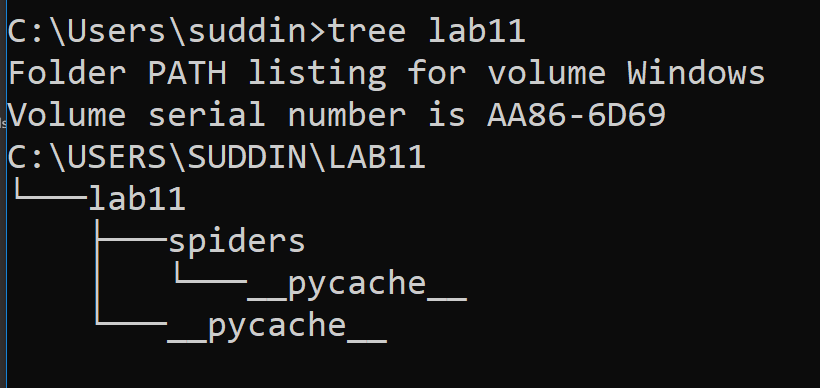


This will create a lab11 directory with some contents:



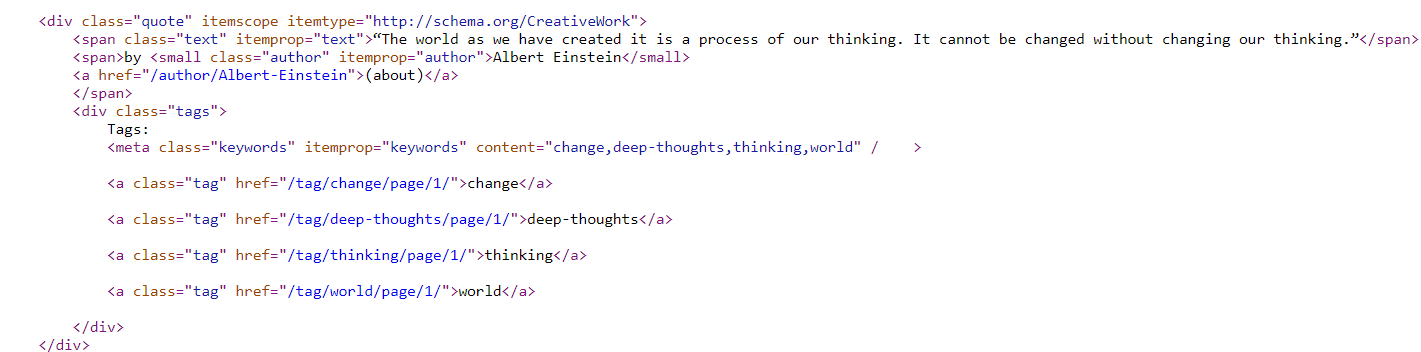
lab11 directory will have following contents:





1. **Writing a spider to crawl a site and extract data**

Here we create a spider by writing the code to extract the quotes from the web page ([http://quotes.toscrape.com](http://quotes.toscrape.com/)). In the source code HTML elements look like following:

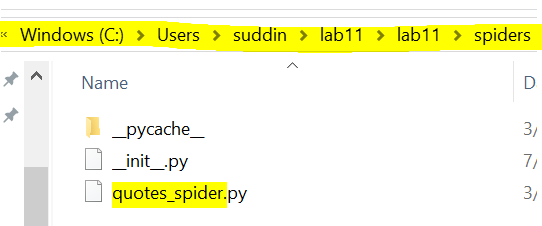


Here is our first code to extract the data from a given URL

|  |
| --- |
| import scrapy  class QuotesSpider(scrapy.Spider):  name = "quotes"  start\_urls = [  'http://quotes.toscrape.com/page/1/',  ]  def parse(self, response):  for quote in response.css('div.quote'):  yield {  'text': quote.css('span.text::text').get(),  'author': quote.css('small.author::text').get(),  'tags': quote.css('div.tags a.tag::text').getall(),  } |
|  |

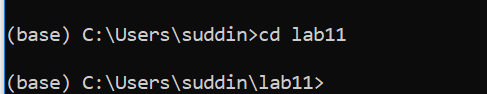
This is code for our first Spider. Save it in a file named **quotes\_spider.py** under the lab11/spiders directory in your project.

***Note:*** *Copy the code in notepad++ or notepad and save it as* ***quotes\_spider.py*** *under the lab11/spiders directory in your project:*



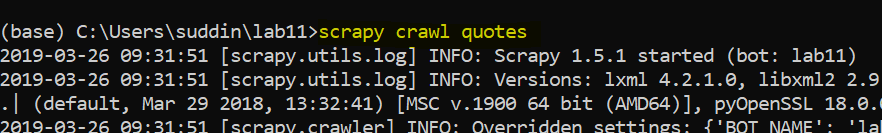
**To Run Spider:**

To run our spider, go to the project’s top level directory and run:

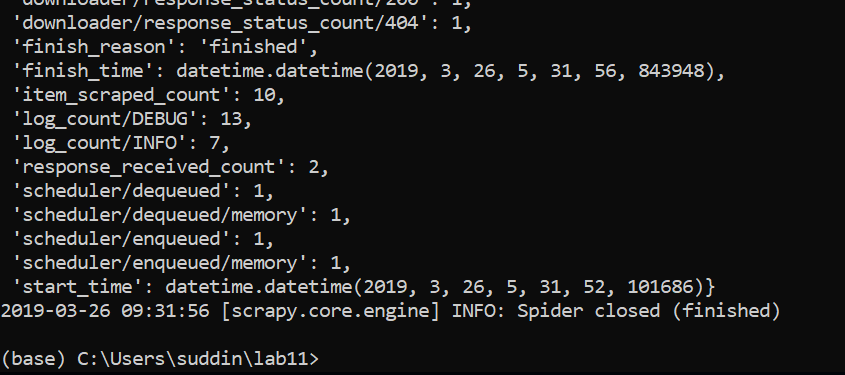


Run the following command:

scrapy crawl quotes



This command runs the spider with name quotes that we’ve just added, that will send some requests for the quotes.toscrape.com domain. You will get an output similar to this:

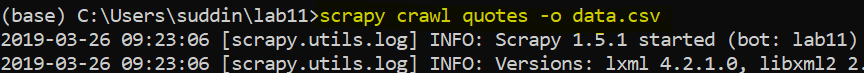


1. **Exporting the scraped data using the command line**

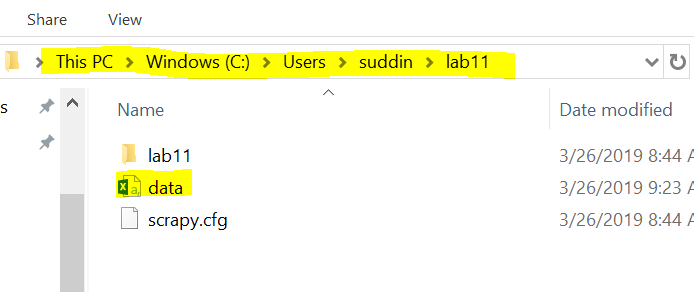
The simplest way to store the scraped data is by using csv file, with the following command:

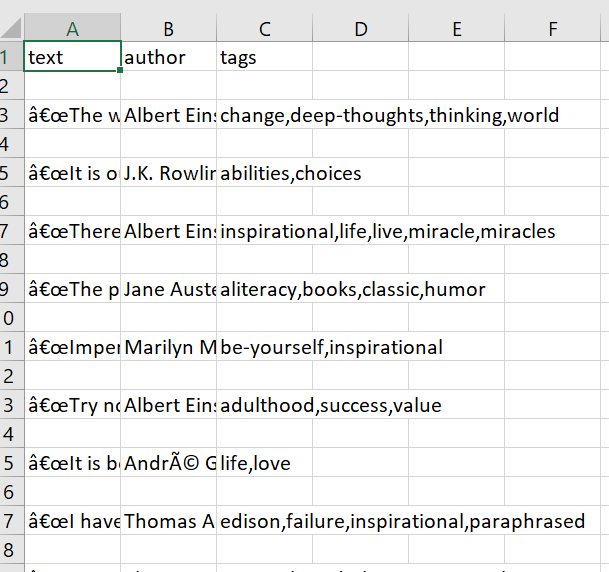
scrapy crawl quotes -o data.csv

Where data.csv is our file to store scrapped data



This will create a file data.csv in your lab11 directory with scrapped data.

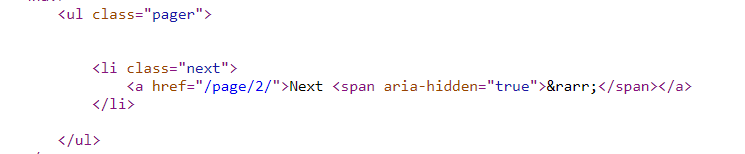




1. **Changing spider to recursively follow links**

Instead of just scraping the stuff from the first two pages from http://quotes.toscrape.com, you want quotes from all the pages in the website.

Now that we have already extracted data from pages, let’s see how to follow links from them. First thing is to extract the link to the page we want to follow. Examining our page, we can see there is a link to the next page with the following markup:

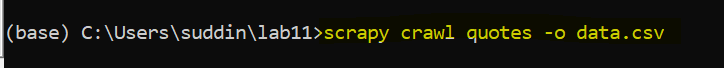


Update the code in quotes.spider.py file to follow the next page link as follow:

|  |
| --- |
| import scrapy  class QuotesSpider(scrapy.Spider):  name = "quotes"  start\_urls = [  'http://quotes.toscrape.com/page/1/',  ]  def parse(self, response):  for quote in response.css('div.quote'):  yield {  'text': quote.css('span.text::text').get(),  'author': quote.css('small.author::text').get(),  'tags': quote.css('div.tags a.tag::text').getall(),  }  next\_page = response.css('li.next a::attr(href)').get()  if next\_page is not None:  next\_page = response.urljoin(next\_page)  yield scrapy.Request(next\_page, callback=self.parse) |
|  |

Now run the following command again and see how many rows of data you have is csv file.

scrapy crawl quotes -o data.csv



1. **Writing a spider to crawl a site and extract data using XPath**

Create a project with name lab12 by repeating step 2 to 5. The only change you have to make is use following code to create quotes.spider.py file instead of the code used in previous example:

Update the code in quotes.spider.py file to follow the next page link as follow:

|  |
| --- |
| import scrapy  class ToScrapeSpiderXPath(scrapy.Spider):  name = 'quotes'  start\_urls = [  'http://quotes.toscrape.com/',  ]  def parse(self, response):  for quote in response.xpath('//div[@class="quote"]'):  yield {  'text': quote.xpath('//span[@class="text"]/text()').extract\_first(),  'author': quote.xpath('//small[@class="author"]/text()').extract\_first(),  'tags': quote.xpath('//div[@class="tags"]/a[@class="tag"]/text()').extract()  }  next\_page\_url = response.xpath('//li[@class="next"]/a/@href').extract\_first()  if next\_page\_url is not None:  yield scrapy.Request(response.urljoin(next\_page\_url)) |
|  |