Webscraping Notes

Chuks Okoli

02 November, 2023

Webscraping

Webscraping is a process that can be used to automatically extract information from a website, and can easily be accomplished within a matter of minutes and not hours.

We import BeautifulSoup. We can store the webpage HTML as a string in the variable HTML. To parse a document, pass it into the BeautifulSoup constructor. We get the BeautifulSoup object, soup, which represents the document as a nested data structure. BeautifulSoup represents HTML as a set of Tree like objects with methods used to parse the HTML. Using the BeautifulSoup object, soup, we created The tag object corresponds to an HTML tag in the original document. For example, the tag "title." Consider the tag <h3>. If there is more than one tag with the same name, the first element with that tag is selected. In this case with Lebron James, we see the name is Enclosed in the bold attribute "b". To extract it, use the Tree representation. Let's use the Tree representation. The variable tag-object is located here. We can access the child of the tag or navigate down the branch as follows: You can navigate up the tree by using the parent attribute. The variable tag child is located here. We can access the parent. This is the original tag object. We can find the sibling of "tag object." We simply use the next sibling attribute. We can find the sibling of sibling one. We simply use the next sibling attribute. Consider the tag child object. You can access the attribute name and value as a key value pair in a dictionary as follows. You can return the content as a Navigable string, this is like a Python string that supports BeautifulSoup functionality.

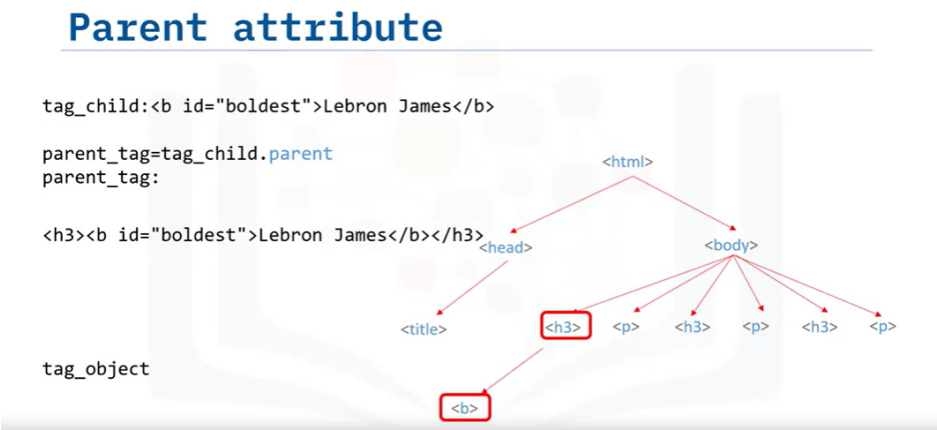

Figure 1: Tree representation

Method find_all

This is a filter, you can use filters to filter based on a tag's name, it's attributes, the text of a string, or on some combination of these. Consider the list of pizza places. Like before, create a BeautifulSoup object. But this time, name it table. The find_all() method looks through a tag's descendants and retrieves all descendants that match your filters. Apply it to the table with the tag <tr>. The result is a Python iterable just like a list, each element is a tag object for <tr>. This corresponds to each row in the list- including

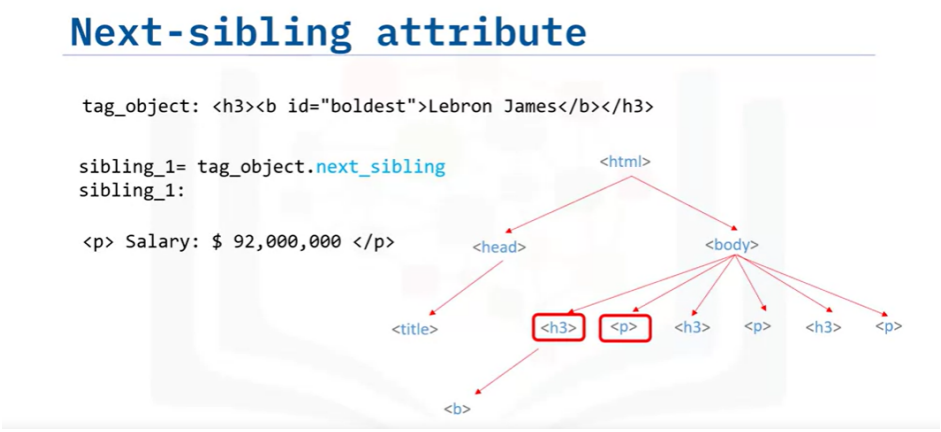

Figure 2: Next-sibling attribute

the table header. Each element is a tag object. Consider the first row. For example, we can extract the first table cell. We can also iterate through each table cell. First, we iterate through the list "table rows," via the variable row. Each element corresponds to a row in the table. We can apply the method find all to find all the table cells, then we can iterate through the variable cells for each row. For each iteration, the variable cell corresponds to an element in the table for that particular row. We continue to iterate through each element and repeat the process for each row. Let's see how to apply BeautifulSoup to a webpage. To scrape a webpage we also need the Requests library. The first step is to import the modules that are needed. Use the get method from the requests library to download the webpage. The input is the URL. Use the text attribute to get the text and assign it to the variable page. Then, create a BeautifulSoup object 'soup' from the variable page. It will allow you to parse through the HTML page. You can now scrape the Page.