Robot Details

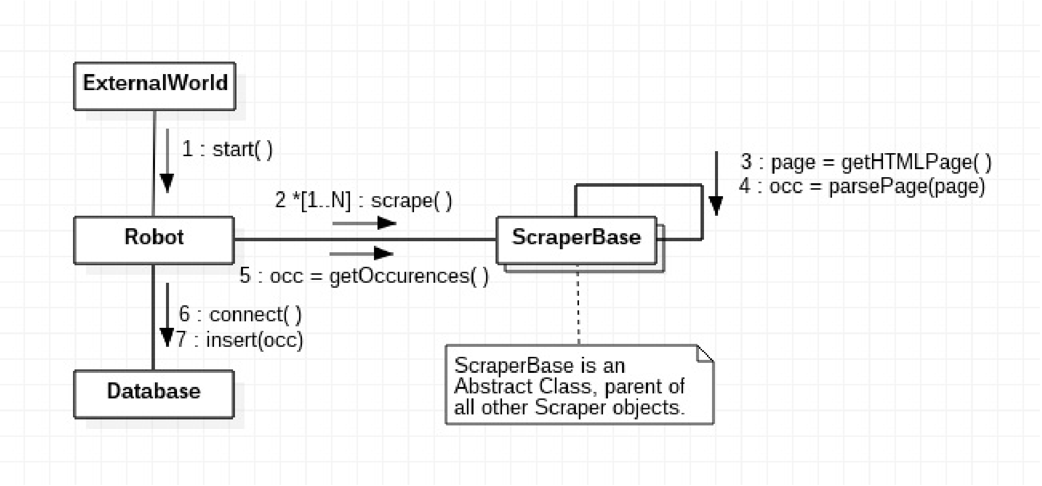
Technologies Used

Whether the web scraping would involve machine learning was in question for some time. The problem with this is that both approaches (with ML and without ML) are completely different in terms of technology demands.

A machine learning approach would require a recursive web scraping technology, provided by Scrapy, while a non-ML approach would have little to no use for such a recursive way of scraping. Hence, my decision was to fully embrace a non-ML approach, and therefore choose technologies accordingly, that is:

- **UrlLib3** for fetching HTML pages from the web
- **BeautifulSoup4** for parsing these pages and getting due data

Scraping Workflow


ExternalWorld
1 : start( )
Robot
2 *[1..N] : scrape( )
5 : occ = getOccurences( )
ScraperBase
3 : page = getHTMLPage( )
4 : occ = parsePage(page)
6 : connect( )
7 : insert(occ)
Database
ScraperBase is an Abstract Class, parent of all other Scraper objects.

As can be seen in the diagram above, the main workflow begins with an external world entity that prompts Robot.py to start. This external world is yet to be decided; it could be periodically every day/hour or it could be through a GET HTTP request in a given port number.

After the start signal, the Robot creates an instance of each Scraper object, which are objects that inherit from ScraperBase. As a child of ScraperBase, each Scraper is supposed to implement the methods `scrape()` and `getOccurences()`.

After instantiating all Scrapers, the Robot runs over the list of Scrapers, calling `scrape()` and then `getOccurences()` on each Scraper and finally appending the received Occurences into Robot’s own list of Occurences.

After getting all Occurences from all Scrapers, the Robot is supposed to send the acquired data somewhere. This is also yet to be decided, but it makes sense to me that Robot inserts the data directly into the database, instead of giving the data to a middle-man that will then do the insertions.