# Data Scraping

## Why it was needed

Data scraping was a necessary part of the team project as we had no access to the data of the SHR website otherwise, we were not given access to the database in anyway. So we had to come up with a data scraping solution to solve this.

## The choices

In order to perform the scraping of data we had to look in to possible choices to achieve the desired result. We looked at a few different methods. The first method that was looked at was using PHP and to scrape the data “by hand” as it was a method used by a team member previously. However due to the amount of data needed and the way in which it was presented, PHP seemed unsuitable for the data scraping. The next choice was JSoup which is a Java library that has the sole purpose of extracting and manipulating data. This was why it was chosen to be used.

## Jsoup

Jsoup([jsoup.org](http://jsoup.org/)) is a java library that was designed in order to allow its users to extract data from websites and manipulate that data. It has a few features that make the act of scraping data relatively simple. It has the basic functions that would be expected of such a library such as the connect function to allow the actual connection to the website. Then there is the select method. The select method is basically a CSS selector that allows the user to specify exactly what piece of HTML/CSS they wish to extract. The website actually features a demo of this([try.jsoup.org](http://try.jsoup.org)). In using the select method we have been able to extract the data that is needed for the team project with reasonably efficient code.

## How it works.

When the program is run the user is presented with a Graphical User Interface with three buttons. Two of which are disabled at first. Upon clicking the start scraping button the user will be presented with output stating how many runners are left to process and when complete the other two buttons are enabled(Update Table and Clear Table). The “Start Scraping” button then becomes a “View Website” button which will take the user to the Westies Runners [website](http://westiesrunners.com).

The program uses Jsoup as explained earlier to select the HTML from the SHR website and is then stored in a relevant CSV file that will be used after the scrape has been completed to update the data to the database for the Westies Runners website.

## Problems Faced.

The first problem faced in the data scraping process was that none of us actually knew or had used Jsoup previously so it had to be learned first. The initial learning period was very short thanks to its ease of use and its focused methods.

One small problem that was faced in the beginning of the project was run time. At that stage we were focusing only on the first 75 or so runners. The scraping program took roughly thirty seconds to extract the relevant data for those runners. While this didn’t seem a huge problem at the time, if it was left as it was it would lead to further problems later. Especially when dealing with a larger amount of data. The problem was solved by using a multi-threaded approach. Instead of visiting each runners page one after the other and dealing with the relevant load times of each we created a thread pool to give out that task as many times as was possible. A CachedThreadPool was used over a FixedThreadPool as it allows more threads to be created as needed and once one has completed its task it can be given the next task. This was chosen as it means the scraping program can run efficiently on different powered machines. A fixed thread pool could make the program run slower than it could potentially run on a more powerful machine. In implementing a thread pool the run has been significantly reduced.

The biggest problem that was faced and took some time to overcome was that on the SHR website the data that was being shown was paginated. The data was paginated in such a way that a simple scrape of the webpage would not acquire all of the needed data from that page as the data was being effectively hidden through a javascript function.

The javascript function that was doing the hiding is known as “\_\_doPostBack” and it is a function used in ASP.NET generated HTML. Its main purpose is to make a large amount of data easier easily human-readable. It works by only showing a select amount of the data to the user at one time and when called again it can show more of the data as needed.

The problem with this function however is that Jsoup cannot simulate a mouse click of a user to allow access to the rest of the data and as none of us really had much experience with javascript it was quite a daunting task to deal with.

The issue was eventually solved by looking at the request headers and form data being sent by the page whenever the javascript was called and using Jsoup to connect to the page whilst sending that data gained.

## Maintainability

As it stands the program is complete as there has been no need for other data at the moment. However if more data is required from the website it would be a reasonably simple process.