COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

COMS W4115 PROGRAMMING LANGUAGES AND TRANSLATORS

# SWIM

Whitepaper

## Team 3

| Name | Role | UNI |
|---|---|---|
| **Morris Hopkins** | Project Manager | mah2250 |
| **Seungwoo Lee** | Language Guru | sl3492 |
| **Lev Brie** | System Architect | ldb2001 |
| **Alexandros Sigaras** | System Integrator | as4161 |
| **Michal Wolski** | Verification and Validation | mtw2135 |

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

## Table of Contents

# Introduction to SWIM

## Inspiration

One of the biggest challenges nowadays in the IT industry is the immense growth in data. According to IDC [1], only in 2011, 1.8 zettabytes of data were generated. To illustrate the size of that, imagine a stack of DVDs spreading from the earth to the moon. By 2020, Gartner [2] predicts that that number will reach 35 zettabytes per year. It is obvious that researchers have access to a sea of information never witnessed before. Can they navigate effectively in that sea or are they often overwhelmed by the scale of information?

## Why is the language needed?

Data scraping is not a new technique and libraries have been developed in the past that can perform data scraping. However, in the absence of a domain specific language, these libraries rely on the syntax provided by the general-purpose-language they are created for. This transforms an otherwise trivial task into an effort that requires otherwise non-trivial programming experience. For many researchers the quest for data is short-lived when faced with this challenge. Such libraries often carry the complexity of the general-purpose language they derive from adding an extra challenge for researchers in the quest for data.

## What do we suggest and who is it for?

We suggest SWIM. SWIM is a domain specific programming language that allows researchers to swim in that sea of data and get exactly what they need. The language targets researchers from all fields with a basic knowledge of Object Oriented Programming and a basic understanding of web development (HTML, JavaScript, CSS). Users of SWIM are not expected to be expert programmers. SWIM is intuitive, expressive and predictable. So, as long as users understand programming concepts they will not drown in the sea of data.

[1] http://www.emc.com/collateral/demos/microsites/emc-digital-universe-2011/index.htm
[2] http://www.marketingtechblog.com/ibm-big-data-marketing/

## What class of problems does SWIM solve?

SWIM solves a problem that many researchers face: data collection and retrieval from the Internet. The idea for SWIM came with the realization that scraping information from webpages is a common problem that people of many backgrounds face on a regular basis. SWIM is NOT a search engine and does not find data. SWIM begins where searching leaves off. Many times researchers know where to find the data and simply need to collect it. SWIM allows users to specify using CSS elements, tags, ids, and classes the sections of a webpage that need to be collected. Whether a user wants to grab a local copy of all the journal entries from a popular site or all the desktop quality wallpapers for a search on Flickr, SWIM makes it easy to do through minimal code, expressive language, and intuitive design. Following are two potential use cases for the SWIM language.

- Meet Jena. Jena is a Masters student in International History in Columbia currently writing her thesis on "Anglo-American relations from world war to cold war". Part of her thesis involves reading historical documents that solidify her approach. Documents that were recently digitized and now reside in a nice digital library in the "cloud". Jena wants to quickly find relative information and download everything neatly in her laptop. Jenna writes a SWIM program that crawls through the journals site looking at the title attributes of all of the article tags. Titles mentioning Anglo-American are downloaded as plain text. In less than day, Jena has all the research material she needs, organized in her laptop and can start working on her thesis from day one.

- A computer vision student is trying to gather thousands of images of leaves from trees so they can run a training program to identify leaf types from images. The student looks up leaf on flickr and gets the following url: http://www.flickr.com/search/?w=all&q=leaf&m=text. With this they write a simple SWIM program that begins to download all of the images that meet the students resolution requirements. From here the student will still need to separate good images from bad but SWIM made collecting his dataset a simple task.

## What are the properties of SWIM?

### Familiar and minimal

SWIM is a web-centric language. Developers gathering information from the web will feel familiar and comfortable with SWIM to achieve their goals. Primitives, operators and iteration functionalities are structured to simplify crawling the web. Additionally, SWIM uses css-selectors to minimize syntax structure when parsing information. This allows developers to filter and produce customized output data, while discarding unwanted information.

### Expressive

The syntax of the SWIM will correspond to words used in natural language. The naming conventions, language operators, and data types will allow developers to accurately express functions in a way that feels natural.

### Extensible

Many researchers and users that need to scrape data from web will write programs that collect information from the same websites. With this in mind SWIM will allow programmers to create packages for specific sites to make it easier for others to collect information from the those sites. For example, a SWIM developer may write a program for collecting articles from JSTOR when accessing the site through a university network. This developer can create package for collecting journal articles from JSTOR that other developers may utilize for their own data collection.

### Object-Oriented

SWIM is object focused. This allows developers to encapsulate individual pieces of their programs so that the whole may be understood as the sum of its parts. Many programs may be concise and have no need of the object model (this keeps the bar low for development), but for more complex data collection objects help keep development clean, and easy to understand.

## What kind of languages exist already and how is our language different?

Currently there are no domain specific languages that tackle this problem. As a result the libraries and tools that exist are dependent on the syntax of the general-purpose programming languages to which they belong. Many experienced programmers are able to use external packages or libraries in a language of their choice but for people who want to focus on data crawling there is tremendous benefit in having a language with intuitive syntax that can be written without the use of auxiliary tools. SWIM responds to this need. Below is a table of resources that exist for several popular programming languages. The number and variety of resources reflect the popularity of the problem at hand. The flavor of the moment seems to be Beautiful Soup, which is a Python package. Other options include Nokogiri for Ruby, Html Agility Pack for .NET, WWW::Mechanize for Perl and Tag Soup for Java.

| Programming Languages | Examples | | | | | | |
|---|---|---|---|---|---|---|---|
| **Python** | Beautiful Soup | lxml | HTQL | Scrapy | Mechanize | | |
| **Ruby** | Nokogiri | Hpricot | Mechanize | scrAPI | scRUBYt! | wombat | Watir |
| **NET** | Html Agility Pack | | | | | | |
| **Perl** | WWW::Mechanize | | Web-Scraper | | | | |
| **Java** | Tag Soup | HtmlUnit | Web-Harvest | jARVEST | | | |
| **PHP** | htmlSQL | | | | | | |
| **Most of them** | Screen-Scraper | | | | | | |
| **Web Service** | Bobik | | | | | | |

Table 1 - Programming Languages with Web Scrapping