# Webscraping in R

Matthew Malishev[1]*
[1] *Department of Biology, Emory University, 1510 Clifton Road NE, Atlanta, GA, USA, 30322*

## Contents

Date: 2018-11-12
R version: 3.5.0
*Corresponding author: matthew.malishev@gmail.com
This document can be found at https://github.com/darwinanddavis

R session info

```
params$session
```

```
R version 3.5.0 (2018-04-23)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: OS X El Capitan 10.11.6

Matrix products: default
BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

loaded via a namespace (and not attached):
 [1] compiler_3.5.0  backports_1.1.2 magrittr_1.5    rprojroot_1.3-2 tools_3.5.0     htmltools_0.3.6
 [7] pillar_1.2.3    tibble_1.4.2    yaml_2.2.0      Rcpp_0.12.19    stringi_1.2.3   rmarkdown_1.10
[13] knitr_1.20      stringr_1.3.1   digest_0.6.15   rlang_0.3.0.1   evaluate_0.10.1
```

## Overview

This document outlines useful tools for webscraping with `R`, including how to use xpaths, navigating XML and JSON, and useful `R` packages.

### Install dependencies

```
packages <- c("rgdal","dplyr","zoo","RColorBrewer","viridis","plyr","digitize","jpeg","devtools","image
if (require(packages)) {
    install.packages(packages,dependencies = T)
    require(packages)
}
lapply(packages,library,character.only=T)
```

## Initial steps

1. Search CRAN for packages that access the API for the site for webscraping, e.g. search 'Spotify' for `Rspotify` package.

2. Use `rvest` or `rjson` to scrape.

3. Use `regex` functions to parse text blocks.

## Selecting individual webpage elements

xpath finder: webpage plugin to isolate HTML elements, e.g. tables on webpage and turn them into XML. Once passed to `R`, packages like `rvest` scrape only that XML element.

Use `tidytext` in R to parse and clean scraped text.

## Geocodes

1. Use Google API to get geocode values from webpage.

2. Run xpath expression to isolate the geocode values e.g "XML parse" function.
   ## PubMed Search pubmed in `R` package on CRAN.
   Use the HTML element web app to find which HTML elements of the page you want to scrape, e.g. = link.
   To scrape the pure text of i.e. just abstract text without the author and affiliation text lines, but across multiple web pages, use the XML search home page to search for and apply an xpath term, e.g. 'abstracttextonly'.

### How to programmatically bypass the 'Show more results' button on webpages

1. Use `RSelenium` package.

2. Find the HTML element on the webpage for the 'Show More' button and access this XML path using the package.