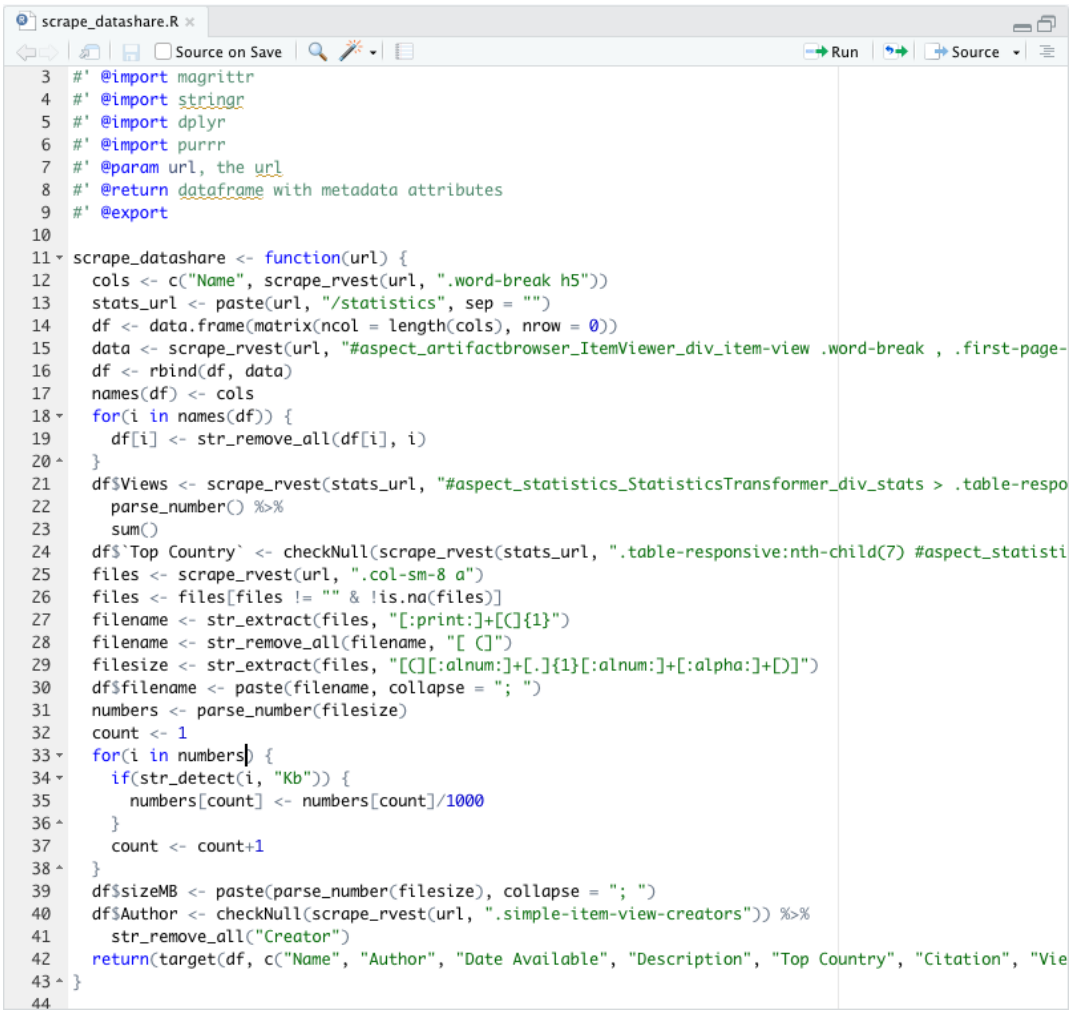
# Metadata Process: Bulleted List

Steps:

1. Identify target attributes you want the user to search by
    - Suggestion: make a list of all variables that you want in your metadata
    - Start with variables that are widely found
    - Classify them as character/categorical variables or numerical variables
2. Find repositories that have at least some of your selected attributes
    - Be sure to check for an API. It can make the scraping process easier and faster
    - The Data Dryad repository has a custom-built R package that uses the existing API to scrape common metadata attributes
3. Scrape the metadata
    - Use a web scraping tool to obtain the text from each website. This can be done with HTML-parsing techniques and CSS tag selectors.
    - The scraping code can vary by repository. It is recommended to write individual functions for each repository. Store the scraped data in a data frame of one row.
    - The code for iterating over the each repository is stored in its own respective file:




- This code uses CSS selectors to extract and clean the text. It is then stored in a data frame with each metadata attribute as a column.

4. Iterate over the repositories
    - Unlike Step 3, the code structure for iterating over the entire repository is quite universal.
    - The code structure is to iterate through the pages of the datasets in the search page of the repository. Parse all dataset links from the webpage, then iterate through such set of links to obtain the metadata.
    - The concept requires a nested for loop with the total number of pages being hard-coded into the loop.
    - Store this data in one data frame per repository.
    - The iterative code structure looks like this:

```r
# Iterate datashare
full_df <- read.csv("./Data/datashare_scraped.csv")
for(i in 3:172) {
  url <- paste("https://datashare.ed.ac.uk/discover?rpp=20&etal=0&group_by=none&page=", i, sep = "")
  link <- Frost2021Package::links(url, ".artifact-description :nth-child(1)")
  link <- link[!is.na(link)]
  link <- paste("https://datashare.ed.ac.uk", link, sep = "")
  for(j in link) {
    print(j)
    print(nrow(full_df))
    full_df <- rbind(full_df, scrape_datashare(j))
    cat("\014")
  }
}
full_df
```

- Since most data repositories organize their data into webpages of data, one would need to iterate over such pages and then go through every dataset on that page. Thus. A nested loop is needed.
- The general structure is to parse the links for each page in a single for loop, then nest another for loop to scrape the metadata and add it to the repository's metadata data frame.
- The print statements are optional; they are for debugging purposes.

5. Combine the scraped data
    - The general process is to "row-bind" all the separated data for each metadata repository, resulting on one data frame.
    - Since each data frame's columns differ from one another, the differing columns in each data frame should be marked as null
        - To make this easier, write a function that takes a data frame and returns a new data frame with a desired columns. If a column in the target columns is not present in the data frame, this should be NA for the whole data frame.
    - Merge each data frame using this function. The "desired columns" should be every column for each data frame.
6. Clean the metadata
    - A simple concatenation and row-bind of each data frame will result in redundant information. Thus, the larger data frame will need cleaning in order to produce a more streamlined result.
    - The recommended technique for cleaning the metadata is to "merge and drop" redundant columns.
        - For example, columns named "Author" and "author" are essentially identical. One would want to merge one column's values into another column, provided that there is content in one column and there is an open space in the column that is being merged to.
        - It is practically impossible for rows to have values for both of these types of columns. Thus, the values from one column are shifted over to the destination column and the origin column is dropped.

- Use the merge-and-drop technique as needed to clean the data.
- Try to find ways to fill in missing values. For example, the delimiter can be determined from a file type (provided that the file name column exists).
- Deposit your data to an existing metadata repository or build your own website for this purpose.