

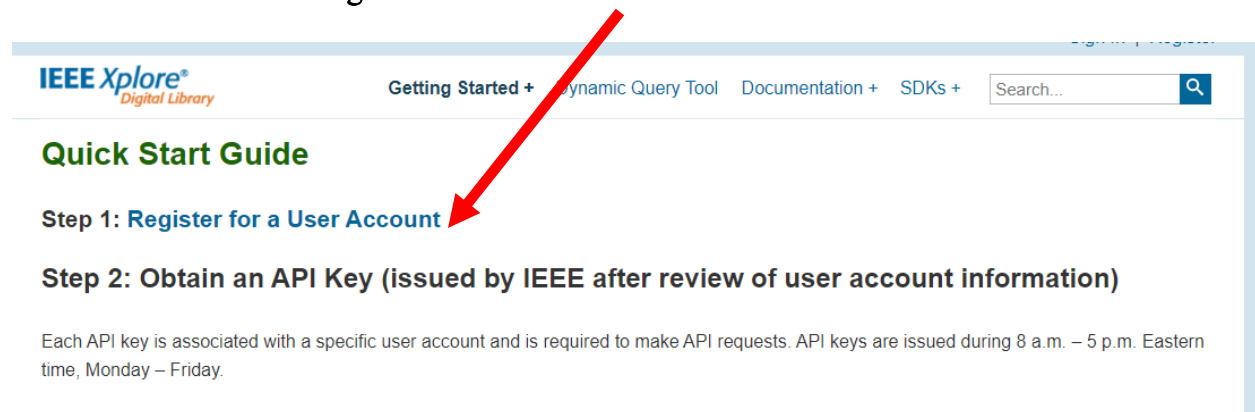
## Lab #9

### Using R to Connect to an API and Retrieve Data

We will use the IEEE Xplore API as an example, but the same principles will work with any API.

**Step # 1** Register and obtain an IEEE Xplore API key (you can skip this step if you already have an IEEE Xplore API key).

Go to: [https://developer.ieee.org/Quick\\_Start\\_Guide](https://developer.ieee.org/Quick_Start_Guide)  
and follow the link to **Register for a User Account**



**Do this today -- it may take a day or two to receive approval-- if you wait until the last day, I will not give you an extension on the deadline.**

- It will ask you for **the url** AND **call back url**, you can use your personal website or mine (<http://www.itk.ilstu.edu/faculty/jrwolf/>) this does not matter because we will only be querying the data. We will not be using it for an application.
- When it asks you the reason for your request, enter:

*I will use the API to research publication frequencies of key terms for a class project.*

**It may take a day or so, but you will receive confirmation of your request and approval via email.**

After you have your key, you may move on to Step 2.

Step 2: Using your key, go to the [Dynamic Query Tool](https://developer.ieee.org/Dynamic_Query_Tool)

[https://developer.ieee.org/Dynamic\\_Query\\_Tool](https://developer.ieee.org/Dynamic_Query_Tool)

and search for articles containing the **Index Term** = “twitter” and **Publication Year** = 2019.  
Set the data format to **JSON**. **Run the query**.

## Pay attention to the results ( this gives the format of the query )

### API Key

Enter Your API Access Key:

### Search & Filter Options

Index Terms	twitter
Publication Year	2019
Select a parameter	

### Result Set Options

Start Position in Results:	1
Maximum Result Set Size:	25
Sort the Results By:	Article Number (Lowest to Highest)
Results Data Format:	JSON

Run Query

Query: [http://ieeexploreapi.ieee.org/api/v1/search/articles?apikey=k...f&format=json&max\\_records=25&start\\_record=1&sort\\_order=asc&sort\\_field=article\\_number&index\\_terms=twitter&publication\\_year=2019](http://ieeexploreapi.ieee.org/api/v1/search/articles?apikey=k...f&format=json&max_records=25&start_record=1&sort_order=asc&sort_field=article_number&index_terms=twitter&publication_year=2019)

### Returns:

```
{
  "total_records": 902,
  "total_searched": 5325232,
  "articles": [
    {
      "doi": "10.1109/TSC.2016.2613048",
      "title": "Memory Partitioning and Management in Memcached",
      "publisher": "IEEE",
      "issue": "4",
      "issn": "2372-0204",
      "rank": 1,
      "volume": "12",
      "authors": {
        "authors": [
          {
            "affiliation": "University of Verona, Verona, Italy",
            "authorUrl": "https://ieeexplore.ieee.org/author/37297140000",
            "id": "37297140000",
            "full_name": "Damiano Carra",
            "author_order": 1
          },
          {
            "affiliation": "EURECOM, Biot, France",
            "authorUrl": "https://ieeexplore.ieee.org/author/37295353100",
            "id": "37295353100",
            "full_name": "Pietro Michiardi",
            "author_order": 2
          }
        ]
      },
      "access_type": "LOCKED",
      "content_type": "Journals",
      "abstract": "Memcached is a popular component of modern Web architectures, which allows fast response times-a fundamental performance index for measuring the Quality of Experience of end-users for serving popular objects. In this work, we study how memory
```

Step # 3 Replicate your Dynamic Query Tool from above query using R

To do this:

a. Open R Studio and install the needed packages

```
install.packages(c("httr", "jsonlite"))
```

Now you are ready to use R to connect to the [IEEE Xplore API](#)

**# b. Load the needed packages**

```
library(httr)
library(jsonlite)
```

**# c. Use the results query to recreate your request**

```
url <- "http://ieeexploreapi.ieee.org/api/v1/search/articles?"
key <- "apikey=k654wxu5huyqs9fxvjm4p9cf"
format <- "&format=json&max_records=25&start_record=1&sort_order=asc&sort_field=article_number"
search.terms <- "&index_terms=twitter&publication_year=2019"
```

**# we will use the paste0 package to pull everything into a single string**

```
z <- paste0(url,key,format,search.terms)
```

**# we will now use the httr package GET function to connect to the IEEE API**  
**# GET is upper case – R is case sensitive**

```
y <- GET(z)
```

```
get_text <- content(y, "text")
```

**# check the output to see that we obtained data**

```
get_text
```

**# convert the JSON data to a data frame**

```
get_text_json <- fromJSON(get_text, flatten = TRUE)
get_text_df <- as.data.frame(get_text_json)
```

**# check the output to see our new dataframe**

```
str(get_text_df)
head(get_text_df$articles.title)
```

#### **Step 4:**

Do steps 2-3 for 4 additional queries

1. affiliation= Illinois State University  
author=Tang
2. publication\_title= IEEE Transactions on Visualization and Computer Graphics  
publication\_year=2018
3. index\_terms=python  
publication\_year=2019
4. index\_terms=javascript  
publication\_year=2019

**Step 5: Use the data from above and `ggplot2` and create a barchart to show which topic (javascript or python) was more popular in 2019.**

Finally, for more additional help using R and API's check out the following article.

<https://www.programmableweb.com/news/how-to-access-any-restful-api-using-r-language/how-to/2017/07/21>

**Submit the R code for steps 4 and 5.**