

Getting Data from the Web with R

Part 8: Getting Data via Web APIs

Gaston Sanchez

April-May 2014

Content licensed under [CC BY-NC-SA 4.0](#)



Readme

License:

Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
<http://creativecommons.org/licenses/by-nc-sa/4.0/>

You are free to:

- Share** — copy and redistribute the material
- Adapt** — rebuild and transform the material

Under the following conditions:

- Attribution** — You must give appropriate credit, provide a link to the license, and indicate if changes were made.
- NonCommercial** — You may not use this work for commercial purposes.
- Share Alike** — If you remix, transform, or build upon this work, you must distribute your contributions under the same license to this one.

Lectures Menu

Slide Decks

1. Introduction
2. Reading files from the Web
3. Basics of XML and HTML
4. Parsing XML / HTML content
5. Handling JSON data
6. HTTP basics and the RCurl package
7. Getting data via Web Forms
8. **Getting data via Web APIs**

Web APIs

Goal

Data via Web APIs

The goal of the present slides is to give you an overview of **how data can be accessed via APIs** with R

Synopsis

In a nutshell

We'll cover the following topics:

- ▶ API basics
- ▶ API considerations
- ▶ Case Study PubMed

Some References

- ▶ XML and Web Technologies for Data Sciences with R
by Deb Nolan and Duncan Temple Lang
- ▶ RESTful Web Services
by Leonard Richardson and Sam Ruby
- ▶ The ROAuth Package
<http://cran.r-project.org/web/packages/ROAuth/index.html>
- ▶ CRAN Task View: Web Technologies and Services
<http://cran.r-project.org/web/views/WebTechnologies.html>

API Basics

API

“In computer programming, an application programming interface (API) specifies how some software components should interact with each other.”

“When used in the context of web development, an API is typically defined as a set of Hypertext Transfer Protocol (HTTP) request messages, along with a definition of the structure of response messages”

http://en.wikipedia.org/wiki/Application_programming_interface

About APIs

What's an API?

API stands for **Application Programming Interface**. Broadly speaking, API refers to a set of programming instructions that allow different software to interact with another.

Web API?

A Web API uses HTTP requests to access information from Web-based software application.

About APIs

What's an API?

By definition, an API is an interface, that is, something that **defines the way in which two software communicate**.

How Web APIs communicate?

With Web APIs, the communication between applications is handled through a collection of standards and protocols (eg HTTP)

Importance

Why should we care?





A lot of companies allow users to freely access data from their websites by means of APIs. So it is important to have some exposure on how to work and get data with APIs.

Everybody who's anybody has an API


Web APIs are becoming increasingly popular for any company, almost mandatory if they want to stay “connected” with the rest of the world, and have a presence with their users/customers. A good starting point for finding more websites with APIs is **Programmable Web**:

<http://www.programmableweb.com>

Web API Examples: Google APIs


    <https://developers.google.com>

Master our APIs and Technologies




Google+

Increase traffic and engagement to your content by using Google+ plugins and APIs.




Android

Use Google APIs in your Android apps.




Cloud Platform

Build and run your websites and apps on Google's infrastructure.




Chrome

Create high performance web apps using the latest technologies the open web has to offer.




Games

Build high powered and interactive web and mobile games using cutting-edge technologies.




Google Maps

The easy way to build interactive data visualizations on a map and location-based apps.




Google Apps

Extend the Google Apps experience for your users.




Google TV

Build apps for the big screen.



Google Wallet

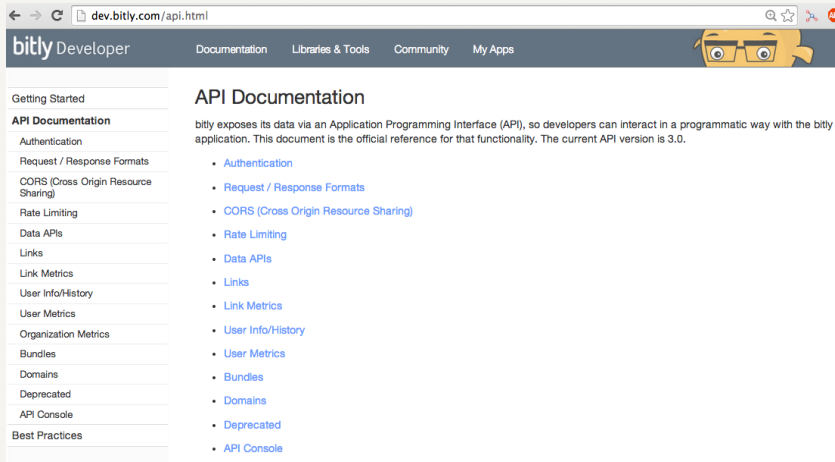
Increase conversions, process payments, and engage customers with offers and loyalty programs.



YouTube

Integrate YouTube's video content and functionality into your website, app, or device.

Web API Examples: Bitly



The screenshot shows a web browser window with the address bar displaying `dev.bitly.com/api.html`. The page has a dark blue header with the "bitly Developer" logo on the left and navigation links for "Documentation", "Libraries & Tools", "Community", and "My Apps" in the center. On the right side of the header is a cartoon character of a yellow notepad with glasses. A left-hand sidebar contains a list of navigation items: "Getting Started", "API Documentation" (which is bolded), "Authentication", "Request / Response Formats", "CORS (Cross Origin Resource Sharing)", "Rate Limiting", "Data APIs", "Links", "Link Metrics", "User Info/History", "User Metrics", "Organization Metrics", "Bundles", "Domains", "Deprecated", "API Console", and "Best Practices". The main content area is titled "API Documentation" and contains a paragraph stating that Bitly exposes its data via an API and that this document is the official reference for version 3.0. Below this paragraph is a bulleted list of links to various API features: Authentication, Request / Response Formats, CORS (Cross Origin Resource Sharing), Rate Limiting, Data APIs, Links, Link Metrics, User Info/History, User Metrics, Bundles, Domains, Deprecated, and API Console.

dev.bitly.com/api.html

bitly Developer

Documentation Libraries & Tools Community My Apps

Getting Started

API Documentation

Authentication

Request / Response Formats

CORS (Cross Origin Resource Sharing)

Rate Limiting

Data APIs

Links

Link Metrics

User Info/History

User Metrics

Organization Metrics

Bundles

Domains

Deprecated

API Console

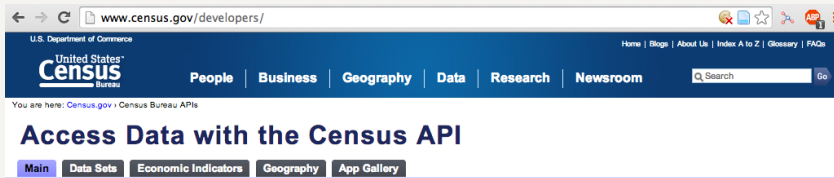
Best Practices

API Documentation

bitly exposes its data via an Application Programming Interface (API), so developers can interact in a programmatic way with the bitly application. This document is the official reference for that functionality. The current API version is 3.0.

- [Authentication](#)
- [Request / Response Formats](#)
- [CORS \(Cross Origin Resource Sharing\)](#)
- [Rate Limiting](#)
- [Data APIs](#)
- [Links](#)
- [Link Metrics](#)
- [User Info/History](#)
- [User Metrics](#)
- [Bundles](#)
- [Domains](#)
- [Deprecated](#)
- [API Console](#)

Web API Examples: US Census



The screenshot shows a web browser at the URL www.census.gov/developers/. The page header includes the U.S. Department of Commerce logo and the United States Census Bureau logo. Navigation links for People, Business, Geography, Data, Research, and Newsroom are present. A search bar is located on the right. Below the header, a breadcrumb trail reads "You are here: Census.gov > Census Bureau APIs". The main heading is "Access Data with the Census API". Below this heading are four tabs: Main, Data Sets, Economic Indicators, and App Gallery. The "Main" tab is currently selected.

Using the API

[Request a Key](#)
[Data Sets](#)

Download Data

To download full data sets,
visit our
[FTP Server](#)

Developer Forum

Need help? Join our [Developer Forum](#) to submit questions, share your apps, and provide feedback.

[Print](#) | [Share this page](#) | [Connect with us](#)

The Census Bureau API

To improve access to data and encourage innovation, the Census Bureau has begun to provide API access to some data sets. We invite developers to use these APIs, join our Developer Forum, and provide feedback to help us move forward with continued API development.

Please read the [Terms of Service](#) for using the API.

Now available: The Economic Indicators Time Series Database - [see details](#)

The Economic Indicator Database is now available via the API. This database includes statistics, primarily at the national level, for measuring key sectors of the U.S. economy.

Discovery Tool (in Beta)

<http://api.census.gov/data.json>

The machine-readable dataset discovery service is now available in beta release. The <http://api.census.gov/data.json> URI produces a JSON document describing all available datasets. The content of this JSON document is based largely on the Open Project Data Common Core Metadata Schema, and extended to include metadata specific to Census Bureau datasets. The <http://api.census.gov/data.xml> URI may be used to access the same information as XML.

In addition to the above URIs dataset discovery is available for the entire vintage/dataset hierarchy as well. For example, the <http://api.census.gov/data/2010.json> and <http://api.census.gov/data/2010.xml> URIs may be used to discover all datasets within the 2010 vintage.

Web API Considerations

Considerations

We're just here for the data

We're interested in Web APIs because they allow us to download content and data. And as with any other tool, there's the good, the bad, and the ugly about Web APIs.

Considerations

The Good

- ▶ Obviously, we get access to data
- ▶ No need to buy or pay for data
- ▶ We can get access in a programmatically way
- ▶ There are some R packages for working with specific Web APIs
<http://cran.r-project.org/web/views/WebTechnologies.html>
- ▶ Under ideal circumstances, data is in nice formats and decent structure

Considerations

The Bad

- ▶ You first have to learn how to use a given Web API
- ▶ Learning about an API and reading the documentation may take you some time
- ▶ The documentation may be very limited, outdated, and poorly explained
- ▶ Sometimes the data is not what you were expecting

Considerations

The Ugly

- ▶ APIs change and evolve, so if you're planning to use one over a long period of time, you have to check it constantly
- ▶ You may need to register, get credentials, use authentication
- ▶ Usually there are rate and quota limits on how much data you can download and how frequently
- ▶ Some APIs can be very complex to use (not user friendly)

Working with Web APIs

Checklist for working with Web APIs

- ▶ Read the API's documentation (check the examples!)
- ▶ Read terms of service (what you're allowed or not to do)
- ▶ Typically you may need an API key
- ▶ You may need to register and open a developer account
- ▶ You may need to authenticate (just another step in the pipeline)
- ▶ See what kind of data you have access to
- ▶ See what kind of formats are available

Working with Web APIs

Working with Web APIs and R

This is our main concern: figure out how to interact with Web APIs and do things in R

- ▶ Understand how the API works
(eg REST, SOAP)
- ▶ Understand what kind of requests you need to use
(GET, POST, etc)
- ▶ Experiment first with simple examples
- ▶ Usually you begin with some kind of query or search
- ▶ Check what R packages you must load
(eg XML, RCurl, SSOAP, ROAuth, httpRequest, RJSONIO)
- ▶ Write functions and scripts (programmatically)

Case Study

E-utilities with PubMed

Case Study Outline


Outline

We'll work with a **small but real life example**, which means you might get lost with all the descriptive information.


Hopefully this brief outline will help you to navigate through this case study:

- ▶ **NCBI Website** (central site containing everything else)
- ▶ Database **PubMed** (this is where we want to get data from)
- ▶ Web API **E-Utilities** (this is the API we'll be playing with)
- ▶ **ESearch** and **ESummary** (in particular, we'll use these 2 *functions*)

National Center for Biotechnology Information

← → ↻ 🔍 ☆  ☰

NCBI Resources How To Sign in to NCBI

 All Databases Search

NCBI Home

Resource List (A-Z)

- All Resources
- Chemicals & Bioassays
- Data & Software
- DNA & RNA
- Domains & Structures
- Genes & Expression
- Genetics & Medicine
- Genomes & Maps
- Homology
- Literature
- Proteins
- Sequence Analysis
- Taxonomy
- Training & Tutorials
- Variation

Welcome to NCBI

The National Center for Biotechnology Information advances science and health by providing access to biomedical and genomic information.

[About the NCBI](#) | [Mission](#) | [Organization](#) | [Research](#) | [NCBI News](#)

Get Started

- [Tools](#): Analyze data using NCBI software
- [Downloads](#): Get NCBI data or software
- [How-To's](#): Learn how to accomplish specific tasks at NCBI
- [Submissions](#): Submit data to GenBank or other NCBI databases

Popular Resources

- PubMed
- Bookshelf
- PubMed Central
- PubMed Health
- BLAST
- Nucleotide
- Genome
- SNP
- Gene
- Protein
- PubChem

NCBI Announcements

NCBI Sequence Viewer version 3.2 available

May 6, 2014

NCBI Sequence Viewer has recently been updated and now has several new features.

Coffee Break tutorial: The promise of

11 1 2 3 4 5 6 7 8

About NCBI

NCBI

The **National Center for Biotechnology Information** (NCBI) provides access to biomedical information

<http://www.ncbi.nlm.nih.gov>

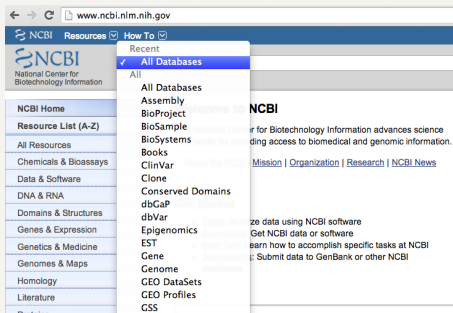
Mind you

- ▶ The NCBI website is kind of ugly and confusing.
(I told you this is a real life example!)
- ▶ Looking at the website, it's not evident what you are supposed to do first or where to begin with.
- ▶ For our example, the important parts of the website are the **Databases** and the **Tools**.

About NCBI

NCBI Databases

The real juice of the NCBI website is the data in it. There are **38 databases** covering a variety of biomedical data. Among them, there is the database *PubMed*.



API: E-Utilities

E-Utilities

The NCBI makes accessible the information in their databases with its API **Entrez Programming Utilities** also known as **E-Utilities**

<http://www.ncbi.nlm.nih.gov/books/NBK25501/>

The screenshot shows the NCBI website's 'Bookshelf' section. The browser address bar displays 'www.ncbi.nlm.nih.gov/books/NBK25501/'. The page title is 'Entrez Programming Utilities Help'. A sidebar on the left contains a book cover for the 'Entrez Programming Utilities Help' manual. The main content area includes a description of the utilities, a search box, and a 'PubReader format: click here to try' button. A right sidebar lists navigation options like 'Views', 'Print View', and 'Cite this Page', as well as 'Other titles in this collection' and 'Recent Activity'.

← → ↻ www.ncbi.nlm.nih.gov/books/NBK25501/ NCBI Resources How To Sign in to NCBI

Bookshelf Books [Search] Browse Titles Limits Advanced Help

Entrez Programming Utilities Help < Prev Next >

Bethesda (MD): National Center for Biotechnology Information (US); 2010-.
Copyright and Permissions

Search this book

PubReader format:
click here to try

Views ▲

PubReader

Print View

Cite this Page

Other titles in this collection ▲

NCBI Help Manual

Related information ▲

NLM Catalog

Recent Activity ▲

Introduction to the E-utilities

- You [E-utilities Introduction](#)
- Please see the [Release Notes](#) for details and changes.

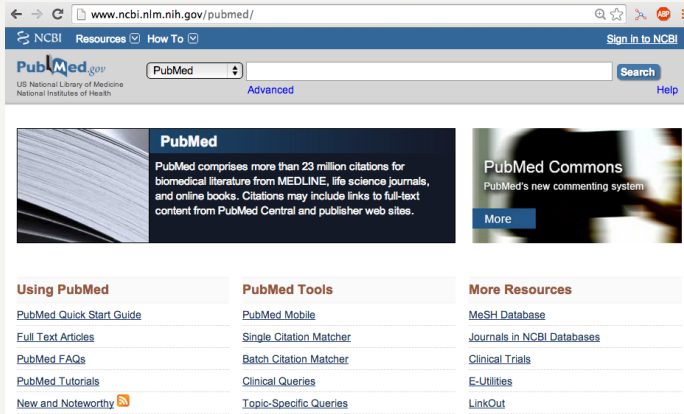
The Entrez Programming Utilities (E-utilities) are a set of eight server-side programs that provide a stable interface into the Entrez query and database system at the National Center for Biotechnology Information (NCBI). The E-utilities use a fixed

About PubMed

PubMed Database

PubMed comprises more than 23 million citations for biomedical literature from MEDLINE, life science journals, and online books.

<http://www.ncbi.nlm.nih.gov/pubmed/>



The screenshot shows the PubMed website homepage. At the top is a navigation bar with the NCBI logo, links to Resources and How To, and a Sign in to NCBI button. Below this is a search bar with the PubMed logo, a dropdown menu set to PubMed, and a Search button. The main content area features a large banner with the PubMed logo and a description of the database. To the right of the banner is a section for PubMed Commons. Below the banner are three columns of links: Using PubMed, PubMed Tools, and More Resources.

← → ↻ www.ncbi.nlm.nih.gov/pubmed/ 🔍 ☆ 🔗 ASP ☰

NCBI Resources ▾ How To ▾ Sign in to NCBI

PubMed.gov
US National Library of Medicine
National Institutes of Health

PubMed ▾ Search


Advanced Help

PubMed

PubMed comprises more than 23 million citations for biomedical literature from MEDLINE, life science journals, and online books. Citations may include links to full-text content from PubMed Central and publisher web sites.

PubMed Commons
PubMed's new commenting system
[More](#)

Using PubMed

- [PubMed Quick Start Guide](#)
- [Full Text Articles](#)
- [PubMed FAQs](#)
- [PubMed Tutorials](#)
- [New and Noteworthy](#) 

PubMed Tools

- [PubMed Mobile](#)
- [Single Citation Matcher](#)
- [Batch Citation Matcher](#)
- [Clinical Queries](#)
- [Topic-Specific Queries](#)

More Resources

- [MeSH Database](#)
- [Journals in NCBI Databases](#)
- [Clinical Trials](#)
- [E-Utilities](#)
- [LinkOut](#)

Searching PubMed

Although we can use the Advanced Search forms to query PubMed (<http://www.ncbi.nlm.nih.gov/pubmed/advanced>), we'll see how to use the E-utilities API to get some data.

The screenshot shows the PubMed Advanced Search Builder web interface. At the top is a navigation bar with 'NCBI', 'Resources', and 'How To' links, along with a 'Sign in to NCBI' button. Below this is a sub-navigation bar with 'PubMed Home', 'More Resources', and 'Help'. The main heading is 'PubMed Advanced Search Builder' with a 'YouTube Tutorial' link. A text box instructs the user to 'Use the builder below to create your search'. There are 'Edit' and 'Clear' links. The 'Builder' section contains two search criteria rows, each with a field selector (currently 'All Fields'), a search term input box, and a 'Show index list' link. The first row has a radio button for 'AND' and the second has a radio button for 'OR'. A 'Search' button and a link to 'Add to history' are at the bottom of the builder section. The 'History' section at the bottom shows 'There is no recent history'.

← → www.ncbi.nlm.nih.gov/pubmed/advanced 🔍 ☆ 🔗 📄 ☰

NCBI Resources How To Sign in to NCBI

PubMed Home More Resources Help

PubMed Advanced Search Builder YouTube Tutorial

Use the builder below to create your search

[Edit](#) [Clear](#)

Builder

All Fields ⌵ [Show index list](#)

AND All Fields ⌵ [Show index list](#)

Search or [Add to history](#)

History

There is no recent history

Mission

Ultimate Goal

Your mission is to download the PubMed IDs for articles published in 2012, and containing the term "*human genome*" in their titles

Good Luck!

Basics of E-utilities

(NCBI Web API)

E-Utilities

E-Utilities provides 8 different applications or utilities:

Application	Description
EInfo	database statistics
ESearch	text searches
EPost	Unique Identification (UID) uploads
ESummary	document summary downloads
EFetch	data record downloads
ELink	Entrez links
EGQuery	global query
ESpell	spelling suggestions
ECitMatch	batch citation searching in PubMed

We'll focus on *ESearch* and *ESummary*

E-Utilities Documentation

Documentation

Like many other Web APIs, there's an extensive documentation about E-Utilities ... and it's not beginner friendly.

But this is a good example of the things you will find when working with APIs. Usually you will struggle reading and understanding the provided documentation.

E-Utilities Applications

How does E-utilities work?

In essence, we use E-utilities by specifying URL's and send requests

1. All E-utilities calls share the same URL:

`http://eutils.ncbi.nlm.nih.gov/entrez/eutils/`

2. Each E-utility has its own structure and parameters:

- ▶ `esearch.fcgi?db=<database>&term=<query>`
- ▶ `esummary.fcgi?db=<database>&id=<uid_list>`

3. All E-utilities are appended to the base url:

`http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=<database>&term=<query>`

ESearch and ESummary

ESearch & ESummary 2-step process

The way you'll work with ESearch and ESummary is using a two-step procedure:

1. you **search for a particular query** in order to get a list of UIDs (unique identification). This is done with **ESearch**
2. you use the list of UIDs to get a summary of the resources you are looking for. This is done with **ESummary**

ESearch

About ESearch

ESearch allows us to specify a **search query on a given database**.
An ESearch call has the following structure:

```
esearch.fcgi?db=<database>&term=<query>
```

Parameters

- ▶ **<database>** represents the name of the database
- ▶ **<query>** represents the terms forming the query

Defining an ESearch Call

Example

We want to search *PubMed* and get a list of *UIDs* for those articles containing the term *human genome* in the title, and that were published in *2012*.

Parameters

database: `pubmed`

query: `human+genome[title]+AND+2012[pdat]`

ESearch Call

`esearch.fcgi?db=pubmed&term=human+genome[title]+AND+2012[pdat]`

ESearch with R

```
# load XML
library(XML)

# E-utilities base url
base_url = "http://eutils.ncbi.nlm.nih.gov/entrez/eutils/"

# define database variable: PubMed
db = "pubmed"

# define query variable
query = "human+genome[title]+AND+2012[pdat]"

# assemble ESearch call
esearch = sprintf("esearch.fcgi?db=%s&term=%s", db, query)

# create URL to make search request
search_url = paste(base_url, esearch, sep = '')

# ESearch request to get the list of UIDs
# (the list will be contained in an XML document)
search_doc = xmlParse(search_url)
```


ESearch XML file output

```
<?xml version="1.0" ?>
<!DOCTYPE eSearchResult PUBLIC ...>
<eSearchResult>
  <Count>103</Count>
  <RetMax>20</RetMax>
  <RetStart>0</RetStart>
  <QueryKey>1</QueryKey>
  <WebEnv>NCID_1_282865897_130.14.18.34_9001_1399488846_694838780</WebEnv>
  <IdList>
    <Id>23959643</Id>
    <Id>23730202</Id>
    ...
  </IdList>
  ...
</eSearchResult>
```

There's a **Count** of 103 articles. Each article has its corresponding **Id**. By default (**RetMax**) just 20 Ids are retrieved

ESearch with R

In order to get the 103 articles, we need to change the default maximum number of retrieved results:

```
# let's add the retmax parameter
retmax = 200

# redefine assemble ESearch
new_esearch = sprintf("esearch.fcgi?db=%s&term=%s&retmax=%s", db, query, retmax)

# update URL to make request
new_search_url = paste(base_url, new_esearch, sep = '')

# ESearch request to get the list of UIDs
# (the list will be contained in an XML document)
new_search_doc = xmlParse(new_search_url)
```

ESearch with R

We need to get the list of `<Id>`'s that are in the XML node `<IdList>`. One way to get those values is by using `xpathSApply()`

```
# get UIDs in a vector
ids = xpathSApply(new_search_doc, path = "//IdList/Id", fun = 'xmlValue')

# we should have 103 IDs
length(ids)

## [1] 103

# take a peek
head(ids)

## [1] "23959643" "23730202" "23520917" "23281708" "23281599" "23266811"

tail(ids)

## [1] "22057813" "22057783" "22050290" "22031941" "21945885" "21559844"
```

ESummary

About ESummary

ESummary allows us to get *document summaries* for a list of input UIDs. An ESummary call has the following structure:

```
esummary.fcgi?db=<database>&id=<uid_list>
```

Parameters

- ▶ `<database>` represents the name of the database
- ▶ `<uid_list>` represents the list of UIDs

ESummary

Example

With ESearch we retrieve the list of UIDs for the articles we're looking for. Now we need to pass the list to ESummary

Parameters

database: `pubmed`

uid_list: `23959643,23730202,23520917,...`

ESummary Call

`esummary.fcgi?db=pubmed&id=23959643,23730202,23520917,...`

ESummary with R

Now we need to use ESummary to get summaries of each article:

```
# concatenate all IDs, separated by commas
# (we'll pass this list to ESummary)
id_list = paste(ids, collapse = ',')

# assemble ESummary call
esummary = sprintf("esummary.fcgi?db=%s&id=%s", db, id_list)

# create URL to make request
sum_url = paste(base_url, esummary, sep = '')

# ESummary request to get document summaries
# (the summaries will be contained in an XML document)
summary_doc = xmlParse(sum_url)
```

ESummary output

You should get something like this from ESummary:

```
## <?xml version="1.0" encoding="UTF-8"?>
## <!DOCTYPE eSummaryResult PUBLIC "-//NLM//DTD esummary v1 20060131//EN" "http://eutils.ncbi.nlm.nih.gov/
## <eSummaryResult>
## <DocSum>
## <Id>23959643</Id>
## <Item Name="PubDate" Type="Date">2012 Dec</Item>
## <Item Name="EPubDate" Type="Date"></Item>
## <Item Name="Source" Type="String">Hum Biol</Item>
## <Item Name="AuthorList" Type="List">
## <Item Name="Author" Type="String">Casto AM</Item>
## <Item Name="Author" Type="String">Henn BM</Item>
## <Item Name="Author" Type="String">Kidd JM</Item>
## <Item Name="Author" Type="String">Bustamante CD</Item>
## <Item Name="Author" Type="String">Feldman MW</Item>
## </Item>
## <Item Name="LastAuthor" Type="String">Feldman MW</Item>
## <Item Name="Title" Type="String">A tale of two haplotypes: the EDA2R/AR Intergenic region is the most c
## <Item Name="Volume" Type="String">84</Item>
## <Item Name="Issue" Type="String">6</Item>
## <Item Name="Pages" Type="String">641-94</Item>
## <Item Name="LangList" Type="List">
## <Item Name="Lang" Type="String">English</Item>
## </Item>
## <Item Name="NlmUniqueID" Type="String">0116717</Item>
## <Item Name="ISSN" Type="String">0018-7143</Item>
```

Journal Names

The summary document of each `<Id>` article is composed of several `<Item>`'s such as "PubDate", "AuthorList", "Title", "Lang", and "FullJournalName", among others.

The **journal name** of each article is in an element:

```
<Item Name="FullJournalName" Type="String">
```

For instance, the first article of the list (`<Id>23959643</Id>`) was published in *Human Biology*:

```
<Item Name="FullJournalName" Type="String">Human  
biology</Item>
```


ESummary with R

If we want to get the journal names in which each article was published, we can use `xpathSApply()` like so:

```
# journal names
journals = xpathSApply(summary_doc, "//Item[@Name='FullJournalName']", xmlValue)

# first 5 journals
head(journals, n = 5)

## [1] "Human biology"
## [2] "Current genomics"
## [3] "Revista de derecho y genoma humano = Law and the human genome review / Catedra de Derecho y Genoma
## [4] "BMC genomics"
## [5] "BMC genomics"

# last 5 journals
tail(journals, n = 5)

## [1] "Human genetics"          "Tissue antigens"
## [3] "Journal of virology"      "Human molecular genetics"
## [5] "Journal of medical systems"
```