Sample Org document

Sampath Singamsetty

December 16, 2023

Contents

Abstract

The document provides the features of org-mode on Emacs and specifically serves as a reference on how to program with multiple langugaes like, emacs-lisp, python, golang, C C++ etc on org-babel. Org mode also provides first hand support for Bibliography and Citations through various addon packages and examples for the same are included.

1 Introduction

This guide features the way `org-babel` interacts with various languages by harnessing the rich type setting features of LaTeX and Emacs. It was originally written on 12-16-2023 using Emacs 29.1 and Org-mode 9.6.6 running on Mac OS X.

This is a very simple org document that uses bibliography, citations and target labels. The next section Sec. 2 uses a `CUSTOM_ID`.

1.1 Applications – uses of integrated code and data

- Reproducible Research (RR)

  > An article about computational science in a scientific publication is not the scholarship itself, it is merely advertising of the scholarship. The actual scholarship is the complete software development environment and complete set of instructions which generated the figures.

- Literate Programming (LP)

  > Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do.

- *working* notes
- executable class notes, presentations and tutorials

2 Citations

A few typical citations:

cite [3]

citeauthor Shannon

citeauthor Kitchin

citenum 3

Table 1: some tabular data.

x	y
0	1

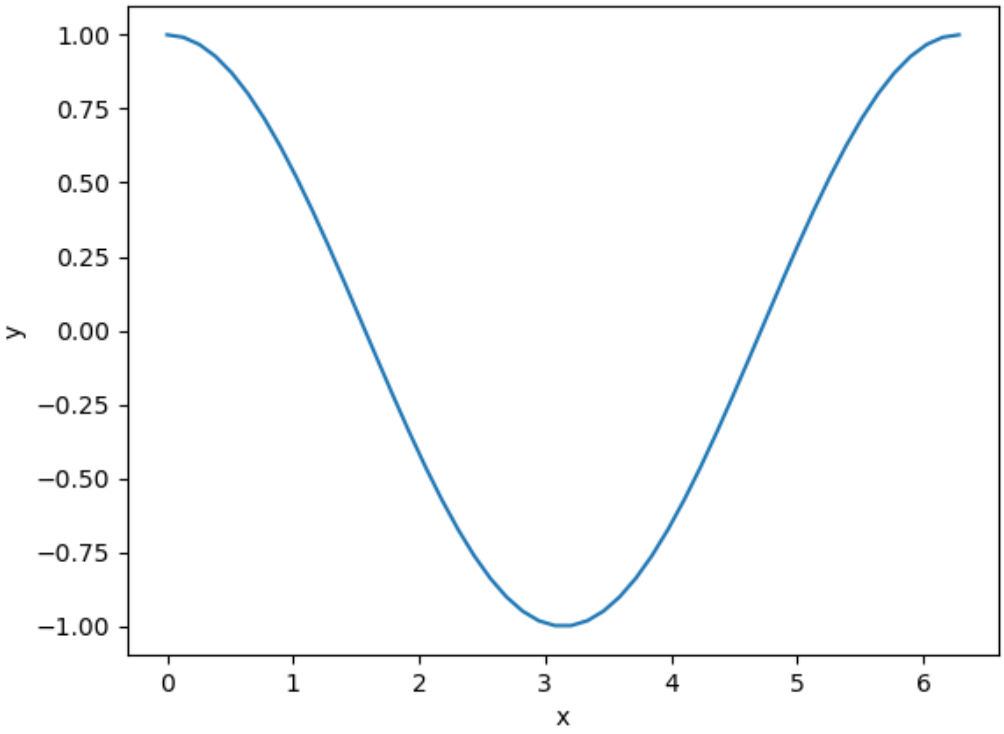


Figure 1: You need a caption.

Here is an equation using a latex label inside it.:

$$\int_0^1 e^x dx \tag{1}$$

$$x_{n+1} = rx_n(1 - x_n)$$

2.1 Cross references

Refer to the section 1. The required data is displayed in the table 1.

Some of the alternate forms of references can be placed as below:

page Fig. 1 is on page 3

name You need a caption (uses the caption)

equation Eq. (1)

With the engagement of cleveref latex package, these cross-references will be modified during compile time so that they have a descriptor in front of each label:

- section 1
- table 1
- fig. 1
- eq. (1)

And here are the same versions in capitalised format:

- Section 1
- Table 1
- Figure 1
- Equation (1)

We may also club several references together like Section 1, table 1, fig. 1, and eq. (1).

And as far as the numerical sequences are concerned, they will be collapsed during compilation to a range: sections 2, 2.1 and 7

The current setup configuration information is in the section Section 2.2.

2.2 Current configuration

Below are the versions of artefacts.

Typing `C-c C-c` (which is the equivalent of pressing `Control` together with `C` twice) on the lines above will call and execute the code blocks declared below. It also works when the cursor is *on* the code blocks.

```
(org-version)
```

```
9.6.6
```

```
(emacs-version)
```

```
GNU Emacs 29.1 (build 1, aarch64-apple-darwin23.0.0, NS appkit-2487.00 Version 14.0 (Build 23A3
 of 2023-10-21
```

```
uname -a
```

```
Darwin Sampaths-MacBook-Pro.local 23.1.0 Darwin Kernel Version 23.1.0: Mon Oct  9 21:27:24 PDT
```

3 Tables

Tables with lots of text in LaTeX often lead to tables that do not fit on a page. This section shows how to produce tables with automatic line breaks.

The `tabularx` latex package has the capability to break lines automatically by using the column specifier X.

Table 2: Temperature, Day & Place

Day	Min Temp	Max Temp	Summary
Monday	11C	22C	A clear day with lots of sunshine. However, the strong breeze will bring down the temperatures
Tuesday	9C	19C	Cloudy with rain, across many northern regions. Clear spells across most of Scotland and Northern Ireland, but rain reaching the far north-west.
Wednesday	10C	21C	Rain will still linger for the morning. Conditions will improve by early afternoon and continue throughout the evening.

3.1 Latex Fonts

LaTeX chooses the appropriate font and font size based on the logical structure of the document (e.g. sections). But in some cases, you may want to set fonts and sizes by hand and this section shows the various font sizes available. To scale text relative to the default body text size, use the commands listed in the table 3

Table 3: LaTeX Font Sizes

Font Size	Text
tiny	Jai Shri Ram
scriptsize	Jai Shri Ram
footnotesize	Jai Shri Ram
small	Jai Shri Ram
normalsize	Jai Shri Ram
large	Jai Shri Ram
Large	Jai Shri Ram
LARGE	Jai Shri Ram
huge	Jai Shri Ram
Huge	Jai Shri Ram

4 Babel source code blocks

4.1 Using shell

```
ls -log
```

```
total 0
drwx------@   6    192 Oct 15 11:03 Applications
drwx------+   3     96 Dec 14 21:10 Desktop
drwx------@   9    288 Nov  2 17:30 Documents
drwx------@  20    640 Dec 16 10:14 Downloads
drwx------@ 100   3200 Oct 19 12:09 Library
drwx------    6    192 Oct  7  2022 Movies
drwx------+   6    192 Mar  9  2022 Music
drwx------+   4    128 Feb  2  2022 Pictures
drwxr-xr-x    3     96 Mar  4  2022 Postman
drwxr-xr-x+   4    128 Feb  2  2022 Public
drwxr-xr-x   35   1120 Dec 11 22:22 aquamacs.d
drwxr-xr-x   28    896 Dec  6 09:41 modules
drwxr-xr-x  102   3264 Nov 27 09:40 scimax
drwxr-xr-x   14    448 Oct  4 10:18 sw
```

4.2 Prologue and Epilogue

Use the `:prologue` and `:epilogue` header arguments to prepare for code blocks to be run, without printing the setup and teardown commands to the exported file. For instance, if the file to be read doesnt exist, create it right before executing the code in the code block, and remove it after.

```
cat -v /tmp/file.txt
```

```
Hello, new file created!
```

4.3 Emacs lisp code

Here is an input data table for `Fibonacci` number generation.

Table 4: fibonacci inputs

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20

Get the version information using `elisp` as below:

```
(princ (concat
        (format "Emacs version: %s\n"
                (emacs-version))
        (format "org version: %s\n"
                (org-version))))
```

Emacs version: GNU Emacs 29.1 (build 1, aarch64-apple-darwin23.0.0, NS appkit-2487.00 Version 14.0 (Build 23A344)) of 2023-10-21 org version: 9.6.6

```
(defun fibonacci (n)
  (if (or (= n 0) (= n 1))
      n
    (+ (fibonacci (- n 1)) (fibonacci (- n 2)))))

(mapcar (lambda (row)
          (mapcar #'fibonacci row)) fib-inputs)
```

1	1	2	3	5	8	13	21	34	55
1	3	8	21	55	144	377	987	2584	6765

Printing some cool numbers

```
(mapcar (lambda (i)
          (list i          (random 10)
                (expt i 2) (random 100)
                (expt i 3) (random 1000)))
        (number-sequence 1 10))
```

1	3	1	46	1	782
2	7	4	17	8	771
3	3	9	20	27	636
4	5	16	84	64	255
5	6	25	78	125	153
6	1	36	38	216	503
7	4	49	37	343	729
8	6	64	67	512	392
9	5	81	1	729	18
10	7	100	59	1000	539

4.4 C & C++ code

Get the installed `gcc` and `g++` versions for running the gnu **C**.

```
gcc --version
```

```
g++ --version
```

Some simple `C` programs outputting some data:

```
printf("mystring %s\n", mystring);
printf("myint    %d\n", myint);
printf("mydouble %g\n", mydouble);
```

mystring	Sunday
myint	81
mydouble	3.14157

4.4.1 Passing the input data to a code block

We can pass a table of data as shown in 5 to the `C++` code block.

Table 5: Sample input data

nb	sqr	noise
zero	0	0.23
one	1	1.31
two	4	4.61
three	9	9.05
four	16	16.55

This table can then be converted to a variable in the script as shown in **??**.

```
#include "stdlib.h"
#include "stdio.h"

int main() {
    for (int i=0; i<somedata_rows; i++) {
        printf ("%2d ", i);
        for (int j=1; j<somedata_cols; j++) {
            const char* cell = somedata[i][j];
            printf ("%5s %5g ", cell, 1000*atof(cell));
        }
        printf("\n");
    }
    return 0;
}
```

```
0      0      0  0.23   230
1      1   1000  1.31  1310
2      4   4000  4.61  4610
3      9   9000  9.05  9050
4     16  16000 16.55 16550
```

```
for(int i = -4; i < 10; i++){
  std::cout << std::setw(10) << std::fixed << std::setprecision(2) << static_cast<double>(i)
            << std::setw(10) << std::fixed << std::setprecision(3) << log(i)
            << "\n";
}
```

-4.0	nan
-3.0	nan
-2.0	nan
-1.0	nan
0.0	-inf
1.0	0.0
2.0	0.693
3.0	1.099
4.0	1.386
5.0	1.609
6.0	1.792
7.0	1.946
8.0	2.079
9.0	2.197

4.5 Special CPP Code blocks

Some of the code blocks might need special libraries like for example the `OpenCV` code block listed in here. It needs the compiled libraries for recognising the OpenCV libraries as shown below and compiling and linking the same.

```
pkg-config --cflags --libs opencv4
```

We can either specify these either with `:libs` value as header or mention them as a `header-args` property for block.

Now the drawer contains a large amount of text, but how much exactly?

```
echo $input
```

```
-L/usr/local/lib -lopencv_gapi -lopencv_stitching -lopencv_alphamat -lopencv_aruco -lopencv_bgs
```

4.6 OpenCV

```
#include "opencv2/core/version.hpp"
#include <opencv2/core.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main() {
    cout << "--- OpenCV Version Information ---" << endl;
    cout << "    OpenCV Version: " << CV_VERSION << endl;
    cout << "    OpenCV version: " << CV_VERSION << endl;
    cout << "    Major version: " << CV_MAJOR_VERSION << endl;
    cout << "    Minor version: " << CV_MINOR_VERSION << endl;
```

```
    cout << "    Subminor version: " << CV_SUBMINOR_VERSION << endl;
}
```

```
--- OpenCV Version Information ---
    OpenCV Version: 4.8.1
    OpenCV version: 4.8.1
    Major version: 4
    Minor version: 8
    Subminor version: 1
```

5 Other C/C++ code blocks

Passing multiple `includes` requires defining them within a list as shown for the next `C++` code block

```
using namespace std;

int main() {
    vector<string> str_vec = {
        "bit", "nibble", "byte",
        "char", "int", "long",
        "long long", "float",
        "double", "long double"
    };

    cout << "--- start ---" << endl;
    for (auto item : str_vec) {
        cout << item << endl;
    }
    cout << "--- done ---" << endl;

    return EXIT_SUCCESS;
}
```

```
--- start ---
bit
nibble
byte
char
int
long
long long
float
double
long double
--- done ---
```

5.0.1 Examples using lambdas

```
int main()
{
  constexpr std::array months{ // pre-C++17 use std::array<const char*, 12>
```

```
    "January",
    "February",
    "March",
    "April",
    "May",
    "June",
    "July",
    "August",
    "September",
    "October",
    "November",
    "December"
  };

  // Search for two consecutive months that start with the same letter.
  const auto sameLetter{
      std::adjacent_find(months.begin(),
                         months.end(),
                         [](const auto& a,
                            const auto& b)
                         {
                                      return a[0] == b[0];
                         })
  };

  // Make sure that two months were found.
  if (sameLetter != months.end())
  {
    // std::next returns the next iterator after sameLetter
    std::cout << *sameLetter << " and " << *std::next(sameLetter)
              << " start with the same letter\n";
  }

  return 0;
}
```

```
June and July start with the same letter
```

5.0.2 Using the noweb syntax

Source code blocks can include references to other source code blocks, using a `:noweb` syntax.

First we define named code blocks

```
void myfunc() {
  printf("print from srcMyfunc\n");
}
```

```
int main(int argc,char **argv) {
  printf("Hello srcMain\n");
  myfunc();
  exit(0);
}
```

Now we define a block that includes the earlier 2 code blocks (which requires the `:noweb` yes option). We could tangle this block, but we can also execute it directly.

```
#include "stdlib.h"
#include "stdio.h"
```

```
void myfunc() {
  printf("print from srcMyfunc\n");
}
int main(int argc,char **argv) {
  printf("Hello srcMain\n");
  myfunc();
  exit(0);
}
```

Hello srcMain print from srcMyfunc Hello srcMain print from srcMyfunc

5.1 GO

golang is supported with the package `ob-go`.

```
fmt.Println("Current Time:", time.Now())
```

```
Current Time: 2023-12-16 12:51:18.750983 +0530 IST m=+0.000138709
```

5.2 Python code

```
from os import listdir

[print(x) for x in listdir('.')]
```

```
basic.bib
references.bib
fig.png
sample.org
obipy-resources
output.png
ltximg
.auctex-auto
example.png
```

We can control whether the output will have horizontal lines or not with the option `:hlines yes/no`. Also whenever the option `:results value` is specified `python` should always return explicitly.

Table 6: input data for python

a	b	c
d	e	f
g	h	i

```
return tab
```

a	b	c
d	e	f
g	h	i

```
return tab
```

a	b	c
d	e	f
g	h	i

If both `:session` and `:results output` are present then the last line should be a function which returns something as shown next.

```
def hello():
    s = 'Hello World'
    return s

hello()
```

```
Hello World
```

Here the body of the code block is implicitly wrapped in a function, the function is called and the return value of the function is the result of the block.

Finally for `:results value` outside a session, use something like below:

```
s = 'Hello World!'
return s
```

```
Hello World!
```

```
import random

random.seed(1)
print("Hello World! Here's a random number: %f" % random.random())
```

```
Hello World! Here's a random number: 0.134364
```

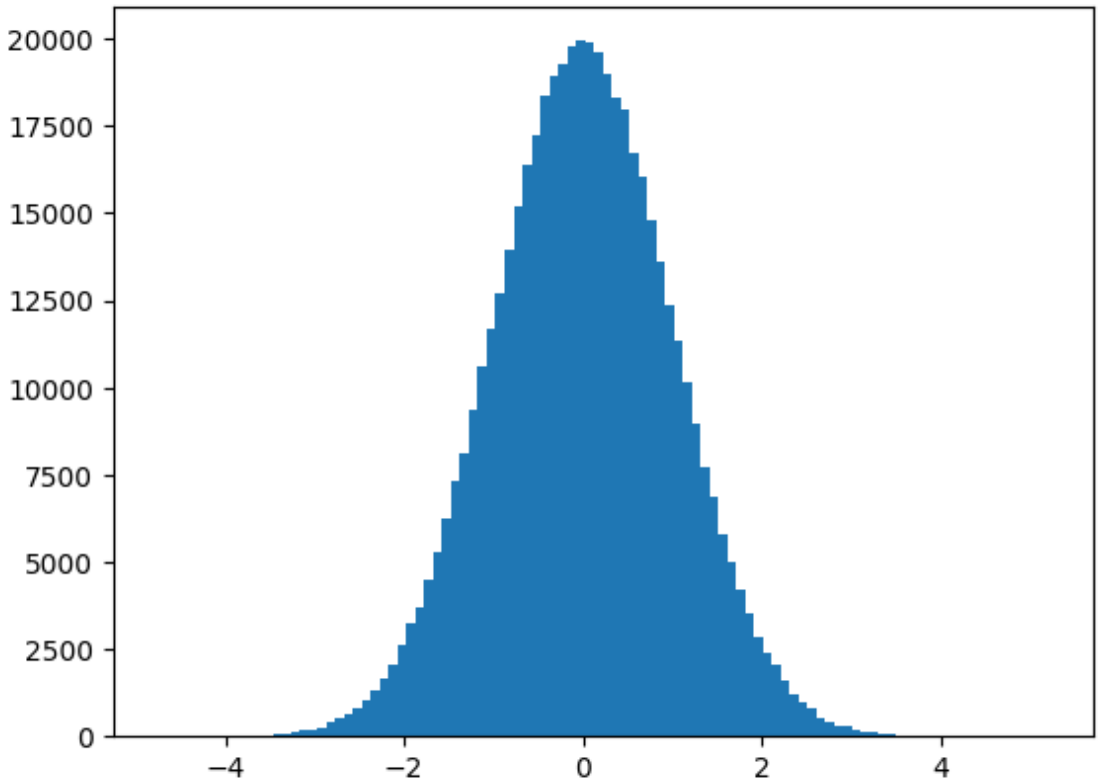
```
def fib(n):    # Write Fibonacci series up to n.
    """ Print a Fibonacci series up to n."""
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
    print()

fib(100)
```

```
0 1 1 2 3 5 8 13 21 34 55 89
```

5.2.1 Using matplotlib

`Matplotlib` is a comprehensive library for creating static, animated, and interactive visualizations in Python. `Matplotlib` makes easy things easy and hard things possible.

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

fig = plt.figure(facecolor='white')
plt.hist(np.random.randn(500000), bins=100)
```

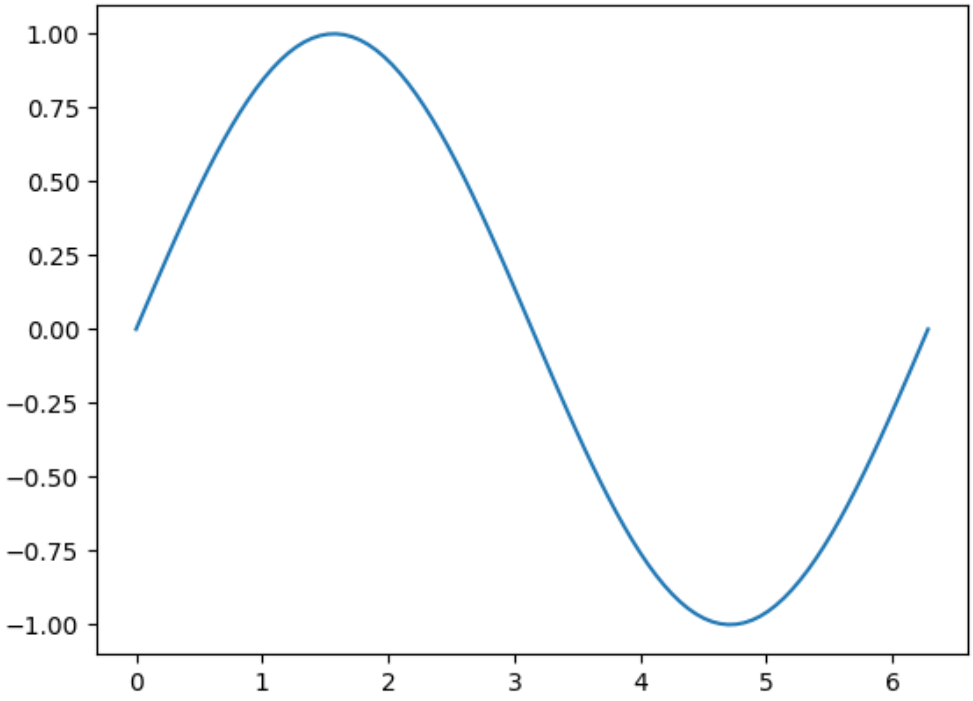



```
%matplotlib inline
%config InlineBackend.figure_format = 'svg'
import matplotlib.pyplot as plt
import numpy as np

# fig, ax = plt.subplots()
# ax.plot([1, 2, 3, 4], [1, 4, 2, 3])
p = plt.hist(np.random.randn(1000), bins=20)
```

[width=.9]./obipy-resources/XRonsh

`plt.show()` doesn't produce output directly, however we can use `:results output file` instead of `:results file` (which is the same as `:results value file`). This means that the data to create the image is taken from `stdout` instead of the value returned directly by python. Because of this we can employ `plt.savefig(sys.stdout.buffer)` to output the image as shown here:

```
import sys
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 2 * np.pi, 100)
y = np.sin(x)
plt.plot(x, y)
plt.savefig(sys.stdout.buffer)
```



5.3 Javascript literate code

```
fetch('https://jsonplaceholder.typicode.com/todos/5')
    .then(x => x.json())
    .then(x => console.log(x));
```

undefined{ userId: 1, id: 5, title: 'laboriosam mollitia et enim quasi adipisci quia provident illum', completed: false }

5.4 Restclient

```
fetch('https://jsonplaceholder.typicode.com/todos/5')
    .then(x => x.json())
    .then(x => console.log(x));
```

```
{
  "userId": 1,
  "id": 10,
  "title": "illo est ratione doloremque quia maiores aut",
  "completed": true
}
// GET https://jsonplaceholder.typicode.com/todos/10
// HTTP/1.1 200 OK
// Date: Sat, 16 Dec 2023 07:21:20 GMT
// Content-Type: application/json; charset=utf-8
// Transfer-Encoding: chunked
// Connection: keep-alive
// Report-To: {"group":"heroku-nel","max_age":3600,"endpoints":[{"url":"https://nel.heroku.com/reports?ts=1702
↪  706298&sid=e11707d5-02a7-43ef-b45e-2cf4d2036f7d&s=SGPXkaAGB%2BaogOwhMk%2FCTdF17ALaTOoSZVb%2BSgLAGhE%3D"}]}
// Reporting-Endpoints: heroku-nel=https://nel.heroku.com/reports?ts=1702706298&sid=e11707d5-02a7-43ef-b45e-2c
↪  f4d2036f7d&s=SGPXkaAGB%2BaogOwhMk%2FCTdF17ALaTOoSZVb%2BSgLAGhE%3D
// Nel: {"report_to":"heroku-nel","max_age":3600,"success_fraction":0.005,"failure_fraction":0.05,"response_he
↪  aders":["Via"]}
// X-Powered-By: Express
// X-Ratelimit-Limit: 1000
// X-Ratelimit-Remaining: 998
// X-Ratelimit-Reset: 1702706327
// Vary: Origin, Accept-Encoding
// Access-Control-Allow-Credentials: true
// Cache-Control: max-age=43200
// Pragma: no-cache
// Expires: -1
// X-Content-Type-Options: nosniff
// Etag: W/"6d-BoXTpHBzlMEesiijbxzpwZqPXhI"
// Via: 1.1 vegur
// CF-Cache-Status: HIT
// Age: 4982
// Server: cloudflare
// CF-RAY: 83652b41ef7b2e9f-HYD
// alt-svc: h3=":443"; ma=86400
// Request duration: 0.069549s
```

5.4.1 Caching data

Calling some API's returns huge data and we can capture it once, and then slice and dice it several ways. This is done by enabling caching. We will put the entire output into a drawer, which can be hidden from view.

```
{
  "userId": 1,
  "id": 10,
  "title": "illo est ratione doloremque quia maiores aut",
  "completed": true
}
// GET https://jsonplaceholder.typicode.com/todos/10
// HTTP/1.1 200 OK
// Date: Sat, 16 Dec 2023 07:21:20 GMT
// Content-Type: application/json; charset=utf-8
// Transfer-Encoding: chunked
// Connection: keep-alive
// Report-To: {"group":"heroku-nel","max_age":3600,"endpoints":[{"url":"https://nel.heroku.com/reports?ts=1702
↪  706298&sid=e11707d5-02a7-43ef-b45e-2cf4d2036f7d&s=SGPXkaAGB%2BaogOwhMk%2FCTdF17ALaTOoSZVb%2BSgLAGhE%3D"}]}
// Reporting-Endpoints: heroku-nel=https://nel.heroku.com/reports?ts=1702706298&sid=e11707d5-02a7-43ef-b45e-2c
↪  f4d2036f7d&s=SGPXkaAGB%2BaogOwhMk%2FCTdF17ALaTOoSZVb%2BSgLAGhE%3D
// Nel: {"report_to":"heroku-nel","max_age":3600,"success_fraction":0.005,"failure_fraction":0.05,"response_he
↪  aders":["Via"]}
// X-Powered-By: Express
// X-Ratelimit-Limit: 1000
```

```
// X-Ratelimit-Remaining: 998
// X-Ratelimit-Reset: 1702706327
// Vary: Origin, Accept-Encoding
// Access-Control-Allow-Credentials: true
// Cache-Control: max-age=43200
// Pragma: no-cache
// Expires: -1
// X-Content-Type-Options: nosniff
// Etag: W/"6d-BoXTpHBzlMEesiijbxzpwZqPXhI"
// Via: 1.1 vegur
// CF-Cache-Status: HIT
// Age: 4982
// Server: cloudflare
// CF-RAY: 83652b41ef7b2e9f-HYD
// alt-svc: h3=":443"; ma=86400
// Request duration: 0.069549s
```

Now the drawer contains a large amount of text, but how much exactly?

```
wc
```

```
1963     4772    62018
```

This last command should have executed instantly, as its working off a cached response from the REST call.

With large responses like this one, it can be hard to get what the overall structure is like, at a glance. Lets use `jq` to create a few summaries of the document.

```
jq 'keys'
```

```
[
  "base_happiness",
  "capture_rate",
  "color",
  "egg_groups",
  "evolution_chain",
  "evolves_from_species",
  "flavor_text_entries",
  "form_descriptions",
  "forms_switchable",
  "gender_rate",
  "genera",
  "generation",
  "growth_rate",
  "habitat",
  "has_gender_differences",
  "hatch_counter",
  "id",
  "is_baby",
  "is_legendary",
  "is_mythical",
  "name",
  "names",
  "order",
  "pal_park_encounters",
  "pokedex_numbers",
  "shape",
  "varieties"
]
```

Now lets look at the first item in the `names` array.

```
jq '.names[10]'
```

```
{
  "language": {
    "name": "zh-Hans",
    "url": "https://pokeapi.co/api/v2/language/12/"
  },
  "name": ""
}
```

We can get a results table by passing the results stream through the @csv filter:

```
jq -r '.names[] | [.name, (.language | .name, .url)] | @csv'
```


```
jq -r '.names[] | [.name, (.language | .name, .url)] | @csv'
```


```
curl -u "someapplication:password" \
     -X GET "http://httpbin.org/basic-auth/someapplication/password" \
     -H "accept: application/json"
```

{ “authenticated”: true, “user”: “someapplication” } { “authenticated”: true, “user”: “someapplication” }

Now, assuming we received the token back we can use the same next

```
jq '.user' | base64
```

```
InNvbWVhcHBsaWNhdGlvbiIK
```

5.5 Example citations

The **Bhagavad Gita** [1] is set in a narrative framework of dialogue between the Pandava prince Arjuna and his charioteer guide Krishna, an avatar of Lord Vishnu.

6 Interesting stuff

Pascal’s Triangle with Lisp and Dot

6.1 Generating Pascal’s Triangle

```
function getPascalsTriangle(n) {
    var arr = {};
    for(var row = 0; row < n; row++) {
        arr[row] = [];
        for(var col = 0; col < row+1; col++) {
            if(col === 0 || col === row) {
                arr[row][col] = 1;
            } else {
```

```
                arr[row][col] = arr[row-1][col-1] + arr[row-1][col];
            }
        }
    }
    return arr;
}

getPascalsTriangle(i)
```

6.2 Converting Pascal's Triangle to Dot with Python

```
console.log('data');
console.log(JSON.parse(data));
```

6.3 Graphing Pascal's Triangle with Dot

7 Miscellaneous

List of Tables

List of Figures

Index

References

[1] Edgerton, F. 1972. *The Bhagavad Gītā*. Harvard University Press.

[2] Kitchin, John R. 2015. Examples of Effective Data Sharing in Scientific Publishing. *ACS Catalysis*, **5**(6), 3894–3899.

[3] Shannon, C. E. 2001. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, **5**(1), 3–55.