

Scraping the web

November 25, 2013 in [HowTo](#), [Scraping](#)



Photo credit: Sharilyn Neidhart

School of Data is re-publishing [Noah Veltman's Learning Lunches](#), a series of tutorials that demystify technical subjects relevant to the data journalism newsroom.

This Learning Lunch is about web scraping: automatically extracting data from web pages.

Web scraping refers to the practice of writing code that will load a web page in order to extract some information from it automatically. It's like a Roomba for the web.

Let's say you wanted to save information about every US federal law with a certain keyword in it. You might have to get it directly from search results in THOMAS. If there are three results, you could just manually copy and paste the information into a spreadsheet. If there are three

thousand results, you probably don't want to do that. Enter the scraper. If you specify exactly what a law looks like in a page of search results, you can then tell the scraper to save every one it finds into a text file or a database or something like that. You also teach it how to advance to the next page, so that even though only ten results are displayed per page, it can cycle through all of them.

When is a scraper useful?

Information you can scrape is virtually always information you can get by hand if you want. It's the same data that gets returned when a human being loads the page. The reason to create a web scraper is to save yourself time, or create an autopilot that can keep scraping while you're not around. If you need to get a lot of data, get data at an automatic interval, or get data at specific arbitrary times, you want to write code that uses an API or a scraper. An API is usually preferable if it supplies the data you want; a scraper is the fallback option for when an API isn't available or when it doesn't supply the data you want. **An API is a key; a scraper is a crowbar.**

See also: [Web APIs](#)

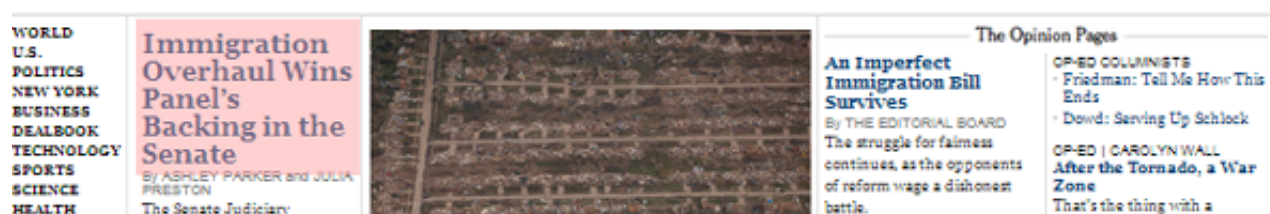
What's hard to scrape?

The best way to understand how a scraper thinks might be to think about what kinds of data are hard to scrape.

Inconsistent data

When it comes to writing a scraper, the name of the game is consistency. You're going to be writing a code that tells the scraper to find a particular kind of information over and over. The first step is detective work. You look at an example page you want to scrape, and probably view the page source, and try to figure out the pattern that describes all of the bits of content you want and nothing else. You don't want false positives, where your scraper sees junk and thinks it's what you wanted because your instructions were too broad; you also don't want false negatives, where your scraper skips data you wanted because your instructions were too narrow. Figuring out that sweet spot is the essence of scraping.

Consider trying to scrape all of the headlines from the New York Times desktop site, excluding video and opinion content. The headlines aren't displayed in the most consistent way, which may make for a better reading experience, but makes it harder to deduce the perfect pattern for scraping:



ARTS
STYLE
OPINION

Autos
Blogs
Books
Cartoons
Classifieds
Crosswords
Dining & Wine
Education
Event Guide
Fashion & Style
Home & Garden
Jobs
Magazine
Media
Movies
Music
Obituaries
Public Editor
Real Estate
Sunday Review
T Magazine
Television
Theater
Travel
Weddings / Celebrations

MULTIMEDIA
Interactives
Photography
Video

TOOL & MORE

Alerts
Beta 620
Corrections
Mobile
Movie Tickets
Learning Network
Newsletters
NYT Events
NYT Store
Theater Tickets
Times Machine
Times Limited
Times Skimmer

Committee, voting 13 to 5 to approve a bill drafted by a bipartisan group of eight senators, agreed to hold off on an amendment that would have added protections for gay couples.

Weiner Announces Candidacy for Mayor in Video

By MICHAEL BARBARO
4:04 AM ET

Anthony D. Weiner, whose political fortunes were derailed by his lewd Internet postings, is trying to restart his career with a run for mayor.

Toile Blasio, Office Shouldn't Be All About Manhattan

Before Apple Tax Breaks, Ireland's Policy Had Critics

By LANDON THOMAS JR. and ERIC PFANNER

Other countries have long been annoyed by Irish laws, which have helped Apple avoid billions of dollars in taxes, but the benefit to the country's economy means the rules are unlikely to change.

Disarming Senators, Apple Chief Eases Tax Tensions

MORE NEWS

Officers Kidnapped in Sinai Are Reported Freed 2:44 AM ET

North Korean Leader Sends Envoy to China 1:52 AM ET

Lobbying Helps Dimon Thwart a Challenge

Iranians Barred From Presidency



Matthew Slaver for The New York Times

A tornado plowed through 17 miles of ground over 50 minutes.

Children Huddled as Storm Sirens Wailed

By MANNY FERNANDEZ and JACK HEALY

Plaza Towers Elementary School's rubble has become the emotional focal point of one of the most destructive tornadoes to strike Oklahoma, which killed 24 on Monday.

Full Graphic: Reading a Scene of Destruction



RELATED COVERAGE

Obama Pledges Storm Aid; Some in Congress Talk of Cuts to Offset It

VIDEO ON THE STORM

Pictures of Loss
Tornado Science: What We Know
A Meteorologist's View of the Tornado
More Video »

INTERACTIVE FEATURE
The Damage in Moore
Sets of 360-degree images show the hard-hit town in Oklahoma before and after.

Storm's Path From the Sky

For Tea Party Groups, Shades of 2010

By TRIP GABRIEL

Leaders of the Tea Party movement hope outrage over the I.R.S. inquiry will rekindle grass-roots activism.

I.R.S. Official Will Decline to Testify Before House Panel

Folk Remedy From Captive Bears Stirs Furor in China

By ANDREW JACOBS

An extract from the gallbladder of bears is believed to have medicinal benefits, but animal welfare advocates aim to convince Chinese consumers of the barbarity of bile farming.

Post a Comment | Read (12) | Video: A Controversial Cure

ON THE BLOGS

Bucks: A \$2,400 Fine for an Airbnb Host

DealBook: Buffett's Touch May Be Irreplaceable

MORE IN OPINION

Bittman: Why I'm Not a Vegan
Op-Ed: China's Brutal One-Child Policy
Op-Doc: 'Sex Offender Village'

tornado — even when you know it's coming, you're helpless, because you don't know precisely where it will hit.

Op-Ed: The View From an Oklahoma Basement

MARKETS »

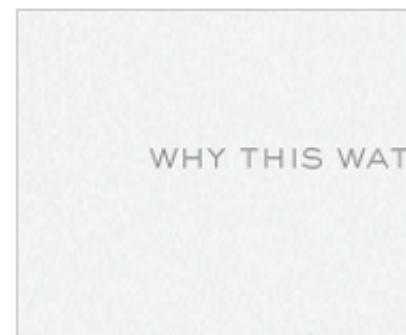
Britain	Germany	France
FTSE 100	DAX	CAC 40
8,792.70	8,467.40	4,026.98
-11.17	-14.80	-10.20
-0.16%	-0.17%	-0.25%

Data delayed at least 15 minutes

GET QUOTES »

My Portfolios »

Stock, ETFs, Funds
Go



WHY THIS WAT

ARTS »

Latest in His Hyphenate? Manager

Quincy Jones, the performer-producer, now 80, has taken on a new role: managing young talent, from jazz musicians to a pan-Asian girl group.

'Arrested' Again

Jessica Walter discusses the new season.

More Interviews: Jason Bateman | Jeffrey Tambor

JUST 99¢
for 4 WEEKS
OF A DIGITAL SUBSCRIPTION

On the other hand, if you looked at the New York Times mobile site, you'd have a comparatively easier time of the same task, because the headlines are displayed with rigid consistency:

World »

Debate Aside, Drone Strikes Drop Sharply

Syrian Forces and Hezbollah Fighters Press Assault on Key City

A Founder of the Revolution Is Barred From Office, Shocking Iranians

Global Business »

Even Before Apple Tax Breaks, Ireland's Policy Had Its Critics

For U.S. Companies, Money 'Offshore' Means Manhattan

Japan Keeps Monetary Policy Steady Amid Deflation Fight

Technology »

Next Xbox Will Face New Array of Rivals

Bits Blog: The New Flickr Is Pretty, but Is It Social?

Bits Blog: An Honor for the Creator of the GIF

Sports »

Global Soccer: Mourinho and Madrid: Loving, Loathing and Leaving

Generally speaking, there are two big tools in the scraper toolbox for how you give a scraper its instructions:

The DOM

A web page is generally made up of nested HTML tags. Think of a family tree, or a phylogenetic tree. We call this tree the DOM.

For example, a table is an element with rows inside of it, and those rows are elements with cells inside of them, and then those cells are elements with text inside them. You might have a table of countries and populations, like this.

```
<table> The whole table
  <tr>
    <td>COUNTRY NAME</td>
    <td>POPULATION</td>
  </tr>
  <tr>
    <td>United States</td>
    <td>313 million</td>
  </tr>
  <tr>
    <td>Canada</td>
    <td>34 million</td>
  </tr>
</table>
```

You will often have the scraper “traverse” this in some fashion. Let’s say you want all the country names in this table. You might tell the scraper, “find the table, and get the data from the first cell in each row.” Easy enough, right? But wait. What about the header row? You don’t

the first cell in each row. Easy enough, right? *But wait.* What about the header row? You don't want to save "COUNTRY NAME" as a country, right? And what if there is more than one table on the page? Just because you only see one table doesn't mean there aren't others. The menu buttons at the top of the page might be a table. You need to get more specific, like "find the table that has 'COUNTRY NAME' in the first cell, and get the data from the first cell in each row besides the first."

You can give a scraper instructions in terms of particular tags, like:

- Find all the `<td>` tags
- Find the second `` tag
- Find every `<a>` tag that's inside the third `<table>` tag

You can also use tag properties, like their IDs and classes, to search more effectively (e.g. "Find every `<a>` tag with the class `external`" would match a tag like `` ").

Regular expressions

Regular expressions are the expert level version of scraping. They are for when the data you want is not neatly contained in a DOM element, and you need to pull out text that matches a specific arbitrary pattern instead. A regular expression is a way of defining that pattern.

Let's imagine that you had a news article and you wanted to scrape the names of all the people mentioned in the article. Remember, if you were doing this for one article, you could just do it by hand, but if you can figure out how to do it for one article you can do it for a few thousand. Give a scraper a fish...

You could use a regular expression like this and look for matches:

```
/[A-Z][a-z]+\s[A-Z][a-z]+/
```

(If you are noticing the way this expression isn't optimized, stop reading, this primer isn't for you.)

This pattern looks for matches that have a capital letter followed by some lowercase letters, then a space, then another capital letter followed by lowercase letters. In short, a first and last name. You could point a scraper at a web page with just this regular expression and pull all the names out. Sounds great, right? *But wait.*

Here are just a few of the false positives you would get, phrases that are not a person's name but would match this pattern anyway:

- Corpus Christi
- Lockheed Martin
- Christmas Eve

- South Dakota

It would also match a case where a single capitalized word was the second word in a sentence, like “The Giants win the pennant!”

Here are just a few of the false negatives you would get, names that would not match this pattern:

- Sammy Davis, Jr.
- Henry Wadsworth Longfellow
- Jean-Claude Van Damme
- Ian McKellen
- Bono
- The Artist Formerly Known as Prince

The point here is that figuring out the right instructions for a scraper can be more complicated than it seems at first glance. A scraper is diligent but incredibly stupid, and it will follow all of your instructions to the letter, for better or worse. Most of the work of web scraping consists of the detective work and testing to get these patterns right.

The simpler and more consistent your information is, the easier it will be to scrape. If it appears exactly the same way in every instance, with no variation, it’s scraper-friendly. If you are pulling things out of a simple list, that’s easier than cherry-picking them from a rich visual layout. If you want something like a ZIP code, which is always going to be a five-digit number (let’s ignore ZIP+4 for this example), you’re in pretty good shape. If you want something like an address, that mostly sticks to a pattern but can have thousands of tiny variations, you’re screwed.

Exceptions to the rule are a scraper’s nemesis, especially if the variations are minor. If the variations are large you’ll probably notice them, but if they’re tiny you might not. One of the dangers in creating a scraper to turn loose on lots of pages is that you end up looking at a sample of the data and making the rules based on that. You can never be sure that the rest of the data will stick to your assumptions. You might look at the first three pages only to have your data screwed up by an aberration on page 58. As with all data practices, constant testing and spot checking is a must.

AJAX and dynamic data

A basic scraper loads the page like a human being would, but it’s not equipped to interact with the page. If you’ve got a page that loads data in via AJAX, has time delays, or changes what’s on the page using JavaScript based on what you click on and type, that’s harder to scrape. You want the data you’re scraping to all be in the page source code when it’s first loaded.

Data that requires a login

Mimicking a logged in person is more difficult than scraping something public. You need to

create a fake cookie, essentially letting the scraper borrow your ID, and some services are pretty sophisticated at detecting these attempts (which is a good thing, because the same method could be used by bad guys).

Data that requires the scraper to explore first

If you want to pull data off a single page, and you know the URL, that's pretty straightforward. Feed the scraper that URL and off it goes. Much more common, though, is that you are trying to scrape a lot of pages, and you don't know all the URLs. In that case the scraper needs to pull double duty: it must scout out pages by following links, and it must scrape the data off the results. The easy version is when you have a hundred pages of results and you need to teach your scraper to follow the "Next Page" link and start over. The hard version is when you have to navigate a complex site to find data that's scattered in different nooks and crannies. In addition to making sure you scrape the right data off the page, you need to make sure you're on the right page in the first place!

Example 1: THOMAS

Let's imagine you wanted to scrape the bill number, title, and sponsor of every law passed by the 112th session of the United States Congress. You could visit [THOMAS](#), the official site for Congressional legislative records. It turns out they keep a [handy list of Public Laws by session](#), so that saves you some time. When you click on the 112th Congress, you'll notice there are links to two pages of results, so you'll need to scrape both of those URLs and combine the results:

Select Congress:

[113](#) | [112](#) | [111](#) | [110](#) | [109](#) | [108](#) | [107](#) | [106](#) | [105](#) | [104](#) | [103](#) | [102](#) | [101](#) →
[View 100-93](#)

[Congress-to-Year Conversion](#)

Select a Range of Public Laws

⌵ [112-1 - 112-150](#)
⌵ [112-151 - 112-283](#)

On each page, you get a list of results that displayed the name and sponsor of the law in a seemingly consistent way:

The LIBRARY of CONGRESS [THOMAS](#)

[The Library of Congress](#) > [THOMAS Home](#) > [Bills, Resolutions](#) > [Search Results](#)

[BACK](#) | [FORWARD](#) | [NEW SEARCH](#) | [HOME](#)

Items **1** through **150** of **283**

Public Laws

1. **H.R.366**: To provide for an additional temporary extension of programs under the Small Business Act and the Small Business Investment Act of 1958, and for other purposes.

Sponsor: [Rep Graves, Sam](#) [MO-6] (introduced 1/20/2011) **Cosponsors** (None)

Committees: House Small Business

Latest Major Action: Became Public Law No: 112-1 [GPO: [Text](#), [PDF](#)]

2. [S.188](#): A bill to designate the United States courthouse under construction at 98 West First Street, Yuma, Arizona, as the "John M. Roll United States Courthouse".

Sponsor: [Sen McCain, John](#) [AZ] (introduced 1/26/2011) [Cosponsors](#) (1)

Committees: Senate Environment and Public Works; House Transportation and Infrastructure

Latest Major Action: Became Public Law No: 112-2 [GPO: [Text](#), [PDF](#)]

3. [H.R.514](#): FISA Sunsets Extension Act of 2011

Sponsor: [Rep Sensenbrenner, F. James, Jr.](#) [WI-5] (introduced 1/26/2011) [Cosponsors](#) (2)

Committees: House Judiciary; House Intelligence (Permanent Select)

Latest Major Action: Became Public Law No: 112-3 [GPO: [Text](#), [PDF](#)]

4. H.J.RES.44 : Further Continuing Appropriations Amendments, 2011

Sponsor: [Rep Rogers, Harold](#) [KY-5] (introduced 2/28/2011) **Cosponsors** (None)

Committees: House Appropriations; House Budget

Latest Major Action: Became Public Law No: 112-4 [GPO: [Text](#), [PDF](#)]

Note: Continuing appropriations through 3/18/2011.

5. H.R.662 : Surface Transportation Extension Act of 2011

Sponsor: [Rep Mica, John L.](#) [FL-7] (introduced 2/11/2011) [Cosponsors](#) (4)

Committees: House Transportation and Infrastructure; House Ways and Means; House Natural Resources; House Budget

House Reports: [112-18](#) Part 1

Latest Major Action: Became Public Law No: 112-5 [GPO: [Text](#), [PDF](#)]

In order to scrape that information and nothing else, we need to put our source code detective hats on (if you're going to start scraping the web, make sure you get a proper source code detective hat and a film noir detective's office). This is where it gets messy. That list looks like this.

[illegible]


```
ep+Graves++Sam))+01656))">Rep Graves, Sam</a>
```

leads us to

```
<a href="/cgi-bin/bdquery/?amp;Db=d112&querybd=@FIELD(FLD003+@4((@1(Rep+Graves++Sam))+01656))">Rep Graves, Sam</a>
```

which leads us to

Rep Graves, Sam

We need to consider special cases. Can a bill have more than one sponsor? Can it have no sponsors? We also need to make sure we parse that name properly. Does every sponsor start with “Rep”? No, some of them are senators and start with “Sen”. Are there any other weird abbreviations? Maybe non-voting delegates from Guam can still sponsor bills and are listed as “Del”, or something like that. We want to spot check as much of the list as possible to test this assumption. We also want to investigate whether a sponsor’s name could have commas in it before we assume we can split it into first and last name by splitting it at the comma. If we scroll down just a little bit on that first page of results, we’ll find the dreaded exception to the rule:

Sen Rockefeller, John D., IV

Now we know that if we split on the first comma, we’d save his name as “John D., IV Rockefeller,” and if we split on the last comma, we’d save his name as “IV Rockefeller, John D.” Instead we need something more complicated like: split the name by all the commas. The first section is the last name, the second section is the first name, and if there are more than two sections, those are added to the end, so we get the correct “John D. Rockefeller IV.”

Example 2: sports teams, stadiums, and colors

Let’s say you wanted to get the list of all NFL football teams with their stadiums, stadium locations, and team colors. Wikipedia has most of this information available in some form or another. You could start [here](#) and you’d find this table:


Image	Stadium	Capacity	Standing Room/Expanded seats	Location	Playing surface	Team(s)	Opened	Ref(s)
	Arrowhead Stadium	76,416	76,416	Kansas City, Missouri	Grass	Kansas City Chiefs	1972	[2]
	Bank of America Stadium	73,778	73,778	Charlotte, North Carolina	Grass	Carolina Panthers	1996	[4]
	Candlestick Park	69,732	69,732	San Francisco, California	Kentucky Bluegrass	San Francisco 49ers	1960 ^[nb 1]	[5]
	CenturyLink Field	67,000	72,000	Seattle, Washington	FieldTurf	Seattle Seahawks	2002	[6]

	Cowboys Stadium†	80,000	105,121	Arlington, Texas	Matrix RealGrass artificial turf ^[7]	Dallas Cowboys	2009	[8]
	Edward Jones Dome†	66,965	66,965	St. Louis, Missouri	AstroTurf GameDay Grass 3D	St. Louis Rams	1995	[9]
	EverBank Field	67,246	76,867	Jacksonville, Florida	419 Tifway Bermuda Grass	Jacksonville Jaguars	1995	[10]
	FedExField	85,000	85,000	Landover, Maryland	419 Tifway Bermuda Grass	Washington Redskins	1997	[11]
	FirstEnergy Stadium	73,200	73,200	Cleveland, Ohio	Kentucky Bluegrass	Cleveland Browns	1999	[12]
	Ford Field†	65,000	70,000	Detroit, Michigan	FieldTurf	Detroit Lions	2002	[13]
	Georgia Dome†	71,228	71,228	Atlanta, Georgia	FieldTurf	Atlanta Falcons	1992	[14]

This seems pretty straightforward. The second column is the stadium name, the fifth column is the location, and the seventh column is the team, so that gets us 3 out of the 4 data points we want.

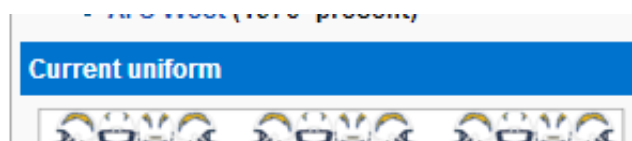
But of course it's not that simple. Wikipedia sometimes includes annotations in the text, and we want to make sure we don't include those as part of what we scrape. Then, if you scroll down to MetLife Stadium, you find the dreaded exception to the rule. It has *two* teams, the Giants and Jets. We need to save this row twice, once for each team.

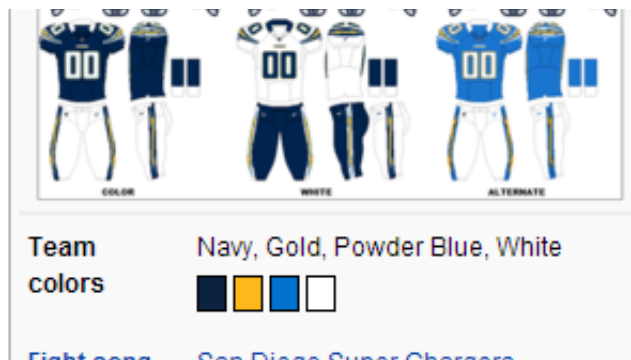
Let's imagine we wanted a more specific location than the city. We could do this by having our scraper follow the link in the "Stadium" column for each row. On each stadium's page, we'd find something like this:

Cowboys Stadium – July 2009	
Location	One Legends Way Arlington, Texas 76011 ^[1] United States
Coordinates	 32°44′52″N 97°5′34″W﻿ / ﻿
Broke ground	September 20, 2005

We could save that and we'd have a precise latitude/longitude for each stadium.

In the same fashion, we can get the team colors by following the link to each team in the "Team(s)" column. We'd find something like this:



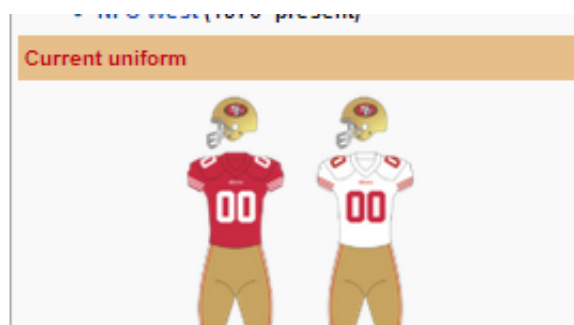


If we inspect those elements, we'd find something like this:

```
▼<tr style="vertical-align: middle;">
  ▼<td>
    <b>Team colors</b>
  </td>
  ▼<td>
    "Navy, Gold, Powder Blue, White"
    ▼<p>
      <span style="background-color:#0C2340; color;; border:1px solid #000000; text-align:center;">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</span>
      <span style="background-color:#FFB81C; color;; border:1px solid #000000; text-align:center;">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</span>
      <span style="background-color:#0072CE; color;; border:1px solid #000000; text-align:center;">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</span>
      <span style="background-color:white; color;; border:1px solid #000000; text-align:center;">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;</span>
    </p>
  </td>
</tr>
```

That “background-color” is what we want, and there’s one for each color swatch on the page. So we could say something like “Find the table cell that says ‘Team colors’ inside it, then go to the next cell and get the background color of every `` inside a pair of `<p>` tags.”

You'll notice that three of them are special color codes ("`#0C2340`", "`#FFB81C`", "`#0072CE`") and one is plain English ("white"), so we need to keep that in mind, possibly converting "white" into a special color code so they all match. As usual, we need to be skeptical of the assumption that every page will follow the same format as this one. If you look at the San Francisco 49ers page, you'll see that the way the colors are displayed varies slightly:



match a particular keyword.

Like

6

g+1



← [Data Analysis for Human Rights Advocacy](#)

[Data Roundup, 26 November](#) →

0 comments



Start the discussion...

Best ▾

Community

Share

Login ▾

No one has commented yet.

— Clip —

• Article

Simplified Article

Full page

Bookmark

Screenshot

— Markup —

ALSO ON SCHOOL OF DATA

[Challenges and Opportunities of Open Data in Moldova](#)

1 comment • 10 days ago



Abayomi Ogundipe — Very interesting article

[OpenRefine/LODRefine for Cleaning Data](#)

1 comment • 5 months ago



Tom Morris — Which be done with standard EU funded the "research"

[At the Cockpit: How the Data Explorer Mission Works](#)

2 comments • 7 months ago



Lucy Chambers — Hi Padraic, Observing current practices - I'm guessing participants are putting in ...

[Events](#)

1 comment • 8 months ago



aa — nice...

— File —

Technical

Add tag

Subscribe

Add Disqus to your site



Written by Noah Veltman

Noah Veltman is a 2013 Knight-Mozilla OpenNews Fellow and a developer/data scientist on the BBC Visual Journalism team. Some of his projects can be found here:
<http://noahveltman.com/sandbox/>

[Share](#)[Save](#)

Get updates from the School of Data in your inbox:

MaryJo Webster

Told a data story recently? Ran a successful data driven campaign? Want to share with others how you did it? **Contact us!** We're always looking for guest posts

Have a data question? Got stuck in a data project? **Ask Schoolofdata!**

On the blog

- Data Expeditions
- Data Stories
- Data Roundup
- Events
- HowTo

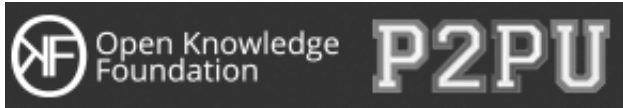
This site uses cookies [More info](#)

☐ Receive announcements ☐ Join the discussion

Get notifications of news from the School in your

inbox

A collaboration between



Built with support from



- [Terms of Use](#)
- [Privacy Policy](#)

All content is licensed under a Creative Commons Attribution-ShareAlike v3.0 License

[Terms of use](#) | [Privacy policy](#) | [IP Policy](#)