**AN XML MIRACLE**

**(Or How to Find a Trader Joe’s That Sells Beer)**

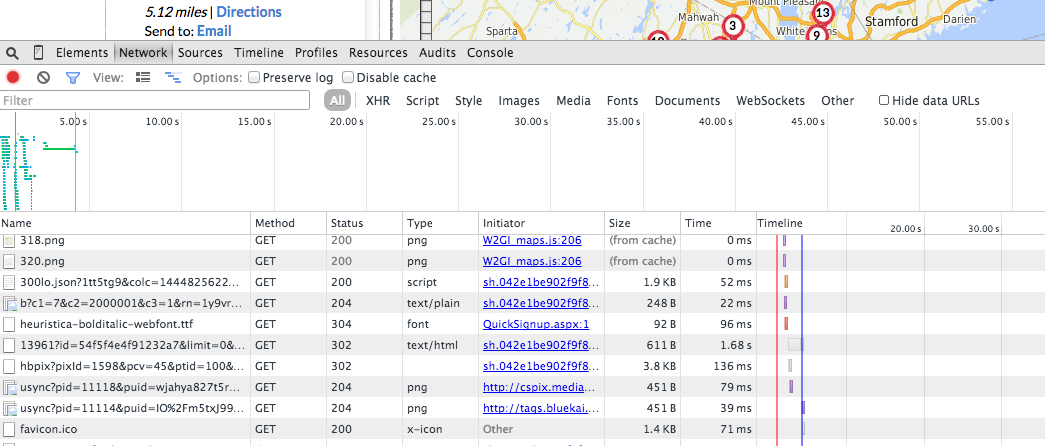
**Robert Gebeloff**

[**rgebeloff@nytimes.com**](mailto:rgebeloff@nytimes.com)

Prof. Susan McGregor, has [a great video](https://www.youtube.com/watch?v=fRYdCwuacyg) on the subject of looking under the hood at data structures we encounter on the Web. It explains how to find underlying data sources that are not immediately obvious, and how these sources are structured – in many cases, a Web page will incorporate data in a format known as XML.

I will leave it to her to explain how XML works, and it’s well worth watching – in this tutorial, we’re going to quickly review how she got the data underlying a Trader Joe’s store search, and then we’re going to convert it to a more Excel-friendly format using a free conversion tool I found by simply Googling “XML converter.”

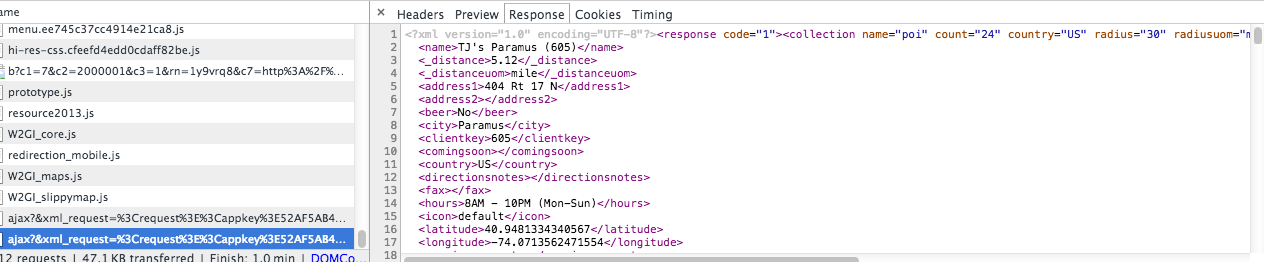
1. Load the TJ’s site: <http://traderjoes.com/stores> (if it doesn’t scarily already know your location, type one in so you have a page that maps and lists the nearby stores)
2. Right click on the results data and “inspect element” – this will open the Chrome inspector, which will help us dig behind the scenes.
3. Click on the Network heading and reload the page. This will generate a list of all the “resources” this page taps into. You should have something that looks like this:



1. Sort these results by “Type” – we’re looking for a resource where the type is a Web data format such as XML or JSON. At the bottom of the list we notice this:



We’re reporters, not programmers. But it looks promising. The name contains xml\_request, and it’s in a format called xhr. We can always look up what that means later (it has to do with how the Prototype javascript library renders data), but right now we just want the data. Click on the last element and you’ll see your prize:



Voila – It’s the data used by the page. It has a structure. But instead of being in rows and columns, it’s in a Webby format – we need to save this out to a file and convert it.

1. Right-click on the element name and Open in a New Tab. This brings our data up in a new browser window along with a long, ugly URL.
2. File | Save Page As and save the data in XML format. It’s important to do it this way instead of copying and pasting what’s on your screen, because you want to preserve all of the underlying XML code that is not being displayed.
3. To see what I mean, open up the XML file in a text editor – the data is wrapped in some code that will tell the XML converter how to handle the file.

The first part tells the putative converter that the data to follow is in a particular XML version. The second part contains meta data about the search that produced the data. And then comes the <poi> data – each one is a store.

<?xml version="1.0" encoding="UTF-8"?><response code="1"><collection name="poi" count="24" country="US" radius="30" radiusuom="mile" centerpoint="-74.0106,40.89" state="NJ" city="Teaneck" address="" province="" postalcode="07666">

<poi>

<name>TJ's Paramus (605)</name>

<\_distance>5.12</\_distance>

<\_distanceuom>mile</\_distanceuom>

<address1>404 Rt 17 N</address1>

<address2></address2>

<beer>No</beer>

<city>Paramus</city>

<clientkey>605</clientkey>

<comingsoon></comingsoon>

<country>US</country>

<directionsnotes></directionsnotes>

<fax></fax>

<hours>8AM - 10PM (Mon-Sun)</hours>

<icon>default</icon>

<latitude>40.9481334340567</latitude>

<longitude>-74.0713562471554</longitude>

<openingcomments></openingcomments>

<phone>201-265-9624</phone>

<postalcode>07652</postalcode>

<province></province>

<specialalcoholhours></specialalcoholhours>

<spirits>No</spirits>

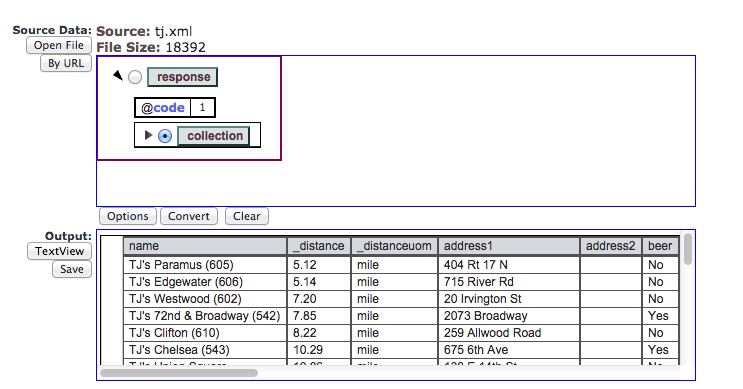
<state>NJ</state>

<uid>1429860907</uid>

<wine>No</wine>

</poi>

1. Let’s use this XML converter I found by Googling. There are others but this one worked well: <http://xmlgrid.net/xml2text.html>
2. Upload the file. Tell it you want the “collection” of data. Convert. It’s a miracle, it seems!



10) Click the Save button. You can now open import the file easily into Excel. Note also how you captured data elements which are not even displayed in the online text, such as the underlying Latitude and Longitude from the map. A good parse indeed!