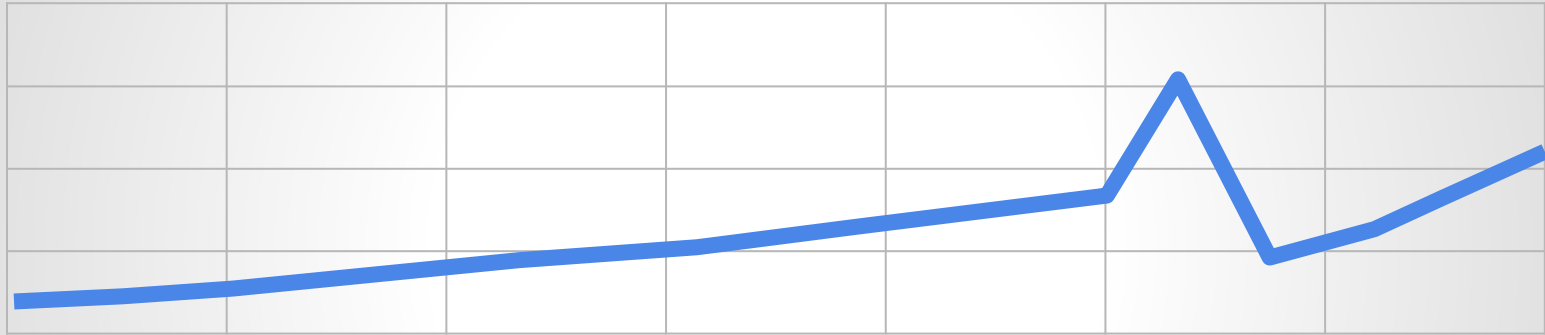


# Connecting charts to live data



Presented at NICAR 2014, Baltimore

Timothy Barmann  
The Providence Journal

[tbarmann@providencejournal.com](mailto:tbarmann@providencejournal.com)

@timothybarmann

**Presentation:** [bit.ly/1cfWL00](http://bit.ly/1cfWL00) **Spreadsheet:** [bit.ly/NcHbHL](http://bit.ly/NcHbHL)

# MAP: Submit snowfall measurements for your Rhode Island town

February 13, 2014 01:26 PM

Comments 0

Share 85 Tweet 239 +1 2 LinkedIn Share 1 Pinterest 8 Email 4



## Example: snowfall map

### Providence Journal reader snowfall report

Your snowfall reports are used to build our map! We'll average the latest three reports from each town and show the results on our color-coded map. Move your mouse over each town to see the snow depth. (If you are having trouble selecting a town, try using your mouse wheel or the arrow buttons on your keyboard.)

\* Required

#### Location \*

Choose the town of your snowfall report

#### Inches \*

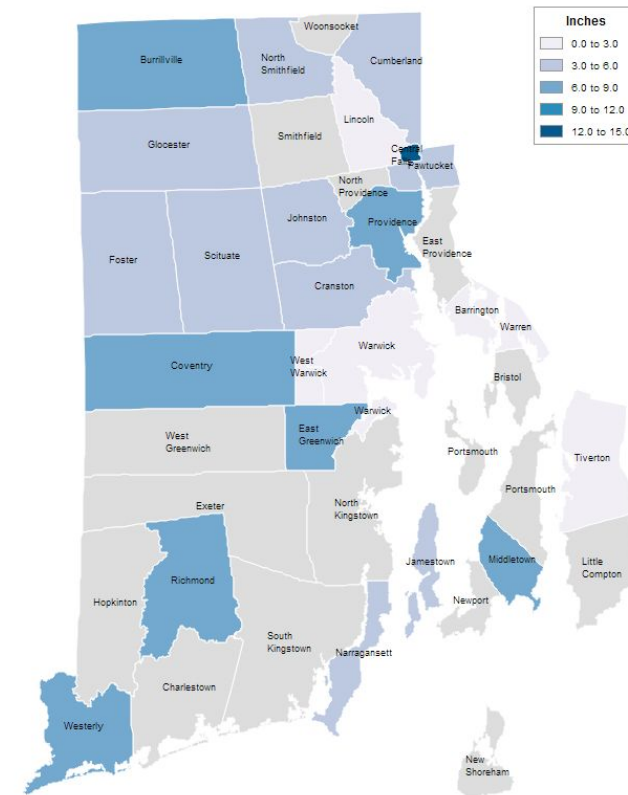
How many inches have accumulated so far

Submit

Never submit passwords through Google Forms.

Powered by This content is neither created nor endorsed by Google.

[Report Abuse](#) - [Terms of Service](#) - [Additional Terms](#)



Total reports: 46  
Last updated: Tue 2/18/2014 3:45 PM

<http://www.providencejournal.com/breaking-news/content/20140213-map-submit-snowfall-measurements-for-your-rhode-island-town.ece>

# Providence Journal reader snowfall report

Your snowfall reports are used to build our map! We'll average the latest three reports from each town and show the results on our color-coded map. Move your mouse over each town to see the snow depth. (If you are having trouble selecting a town, try using your mouse wheel or the arrow buttons on your keyboard.)

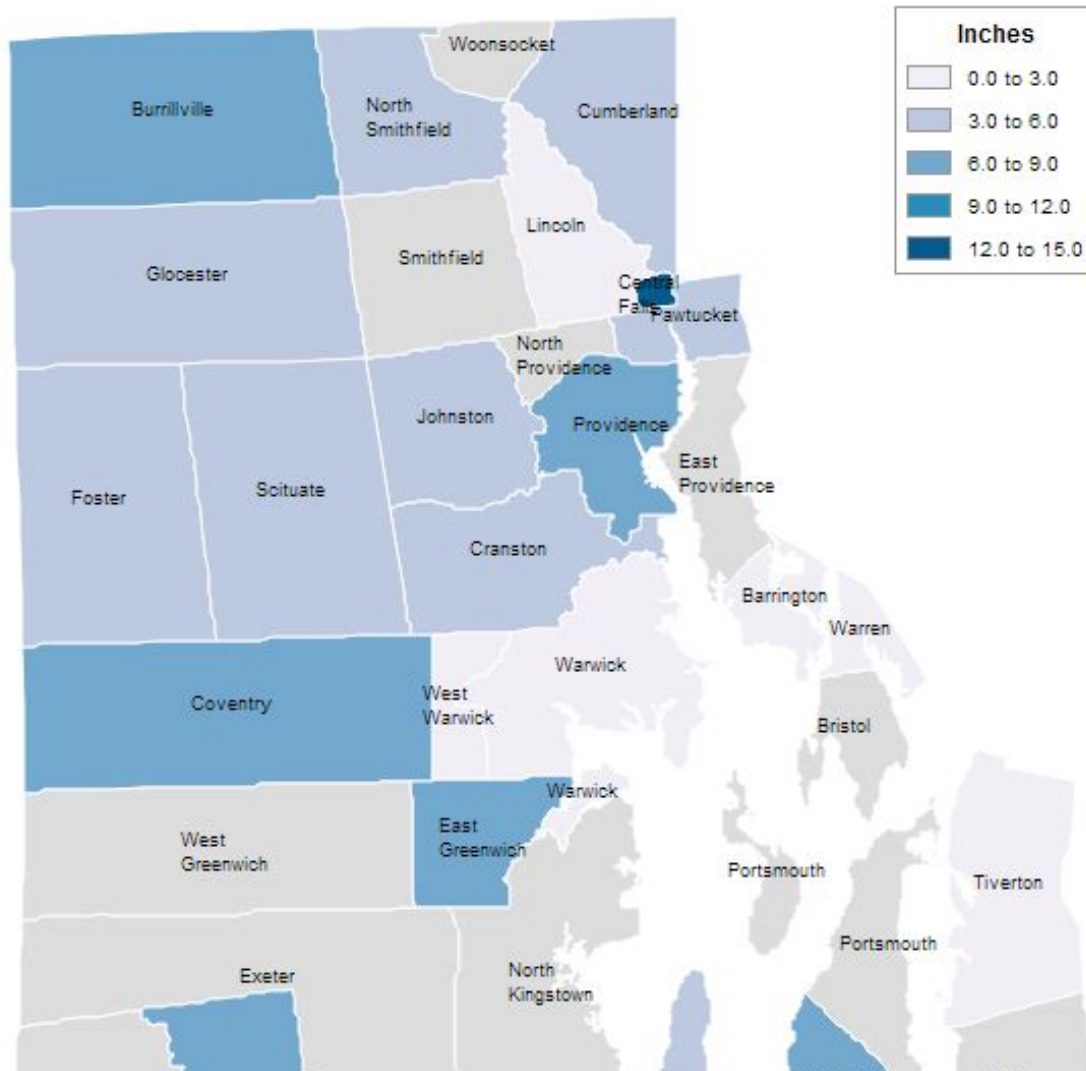
\* Required

## Location \*

Choose the town of your snowfall report

## Inches \*

How many inches have accumulated so far



# Example: fuel prices



Source: [R.I. Office of Energy Resources](#), updated each Friday.

Providence Journal/Timothy C. Barmann

Hold your mouse over chart line to see date and price.

# Example: fuel prices

 State of Rhode Island  
Office of Energy Resources

Search this site:  Go

Home > Gasoline Prices

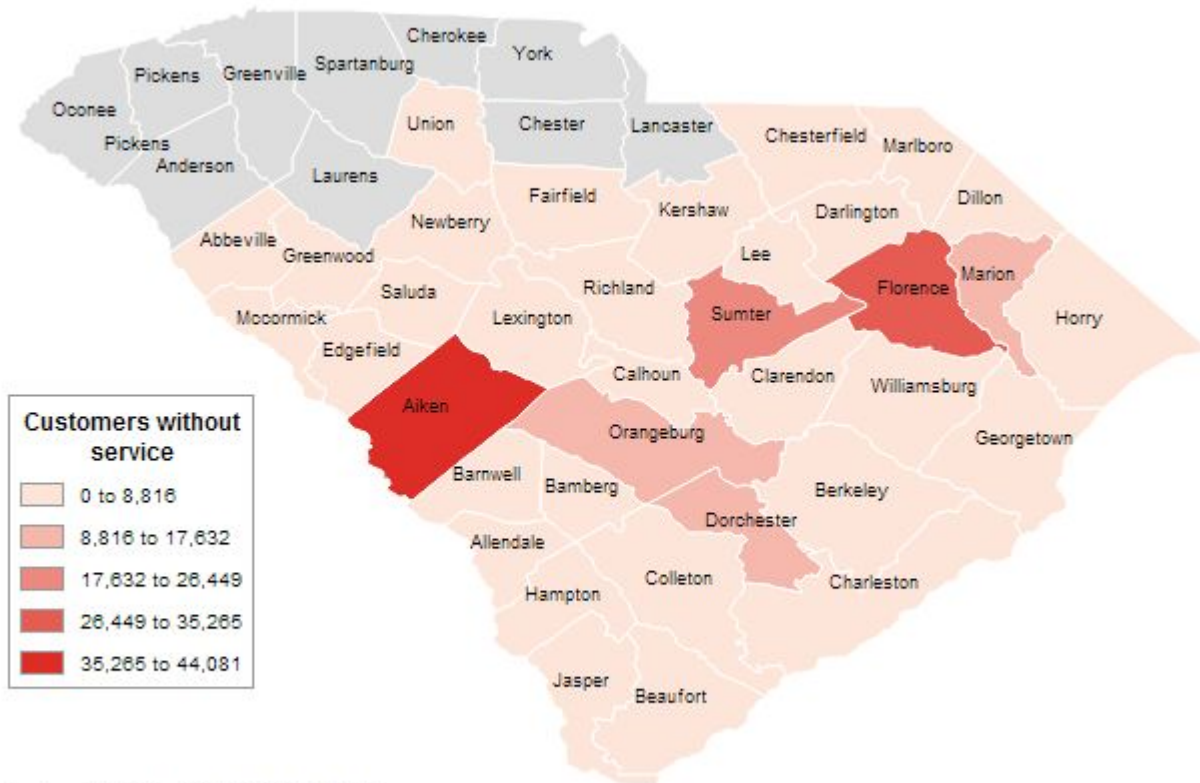
### Gasoline Prices

RHODE ISLAND ENERGY OFFICE - GASOLINE SURVEY

DATE	SELF SERVE AVERAGE			FULL SERVE AVERAGE		
	REGULAR UNLEADED	UNLEADED PLUS	PREMIUM UNLEADED	REGULAR UNLEADED	UNLEADED PLUS	PREMIUM UNLEADED
02/21/2014	\$3.58	\$3.80	\$3.94	\$3.69	\$3.91	\$4.05
02/14/2014	\$3.50	\$3.73	\$3.87	\$3.63	\$3.83	\$3.98
01/31/2014	\$3.48	\$3.71	\$3.86	\$3.59	\$3.85	\$3.98
01/24/2014	\$3.49	\$3.72	\$3.86	\$3.63	\$3.88	\$4.01
01/17/2014	\$3.51	\$3.73	\$3.87	\$3.69	\$3.93	\$4.06
01/10/2014	\$3.54	\$3.77	\$3.91	\$3.76	\$3.98	\$4.11
01/06/2014	\$3.55	\$3.73	\$3.92	\$3.70	\$3.87	\$4.03
12/26/2013	\$3.54	\$3.75	\$3.90	\$3.75	\$3.94	\$4.09
12/19/2013	\$3.52	\$3.74	\$3.88	\$3.65	\$3.84	\$3.96
12/09/2013	\$3.50	\$3.71	\$3.85	\$3.68	\$3.92	\$4.06
12/02/2013	\$3.48	\$3.68	\$3.82	\$3.70	\$3.88	\$4.10
11/21/2013	\$3.40	\$3.62	\$3.85	\$3.60	\$3.77	\$4.00
11/15/2013	\$3.39	\$3.66	\$3.78	\$3.62	\$3.88	\$3.99
11/08/2013	\$3.39	\$3.63	\$3.80	\$3.55	\$3.83	\$3.97
11/01/2013	\$3.42	\$3.68	\$3.68	\$3.83	\$4.04	\$4.23
10/25/2013	\$3.45	\$3.70	\$3.83	\$3.73	\$3.94	\$4.04
10/11/2013	\$3.48	\$3.74	\$3.83	\$3.68	\$3.90	\$4.04

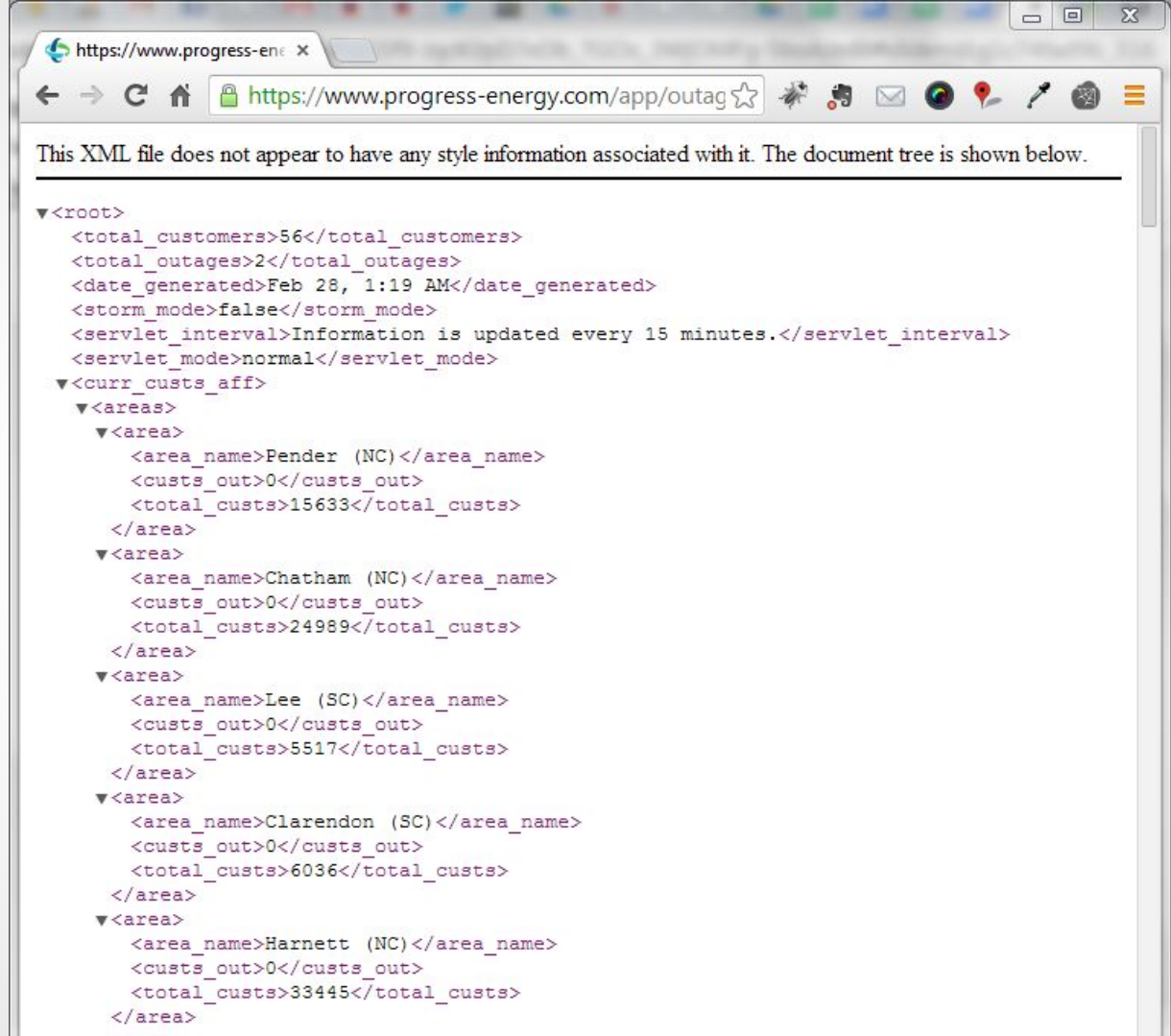
<http://www.energy.ri.gov/energyprices/gasprices.php>

# Example: power outages



Last updated: Thu 2/13/2014 11:55 AM

# Example: power outages



The screenshot shows a web browser window with the address bar displaying `https://www.progress-energy.com/app/outag`. The page content displays an XML document tree. At the top, a message states: "This XML file does not appear to have any style information associated with it. The document tree is shown below." The XML tree is expanded to show the following structure:

```
<root>
  <total_customers>56</total_customers>
  <total_outages>2</total_outages>
  <date_generated>Feb 28, 1:19 AM</date_generated>
  <storm_mode>false</storm_mode>
  <servlet_interval>Information is updated every 15 minutes.</servlet_interval>
  <servlet_mode>normal</servlet_mode>
  <curr_custs_aff>
    <areas>
      <area>
        <area_name>Pender (NC)</area_name>
        <custs_out>0</custs_out>
        <total_custs>15633</total_custs>
      </area>
      <area>
        <area_name>Chatham (NC)</area_name>
        <custs_out>0</custs_out>
        <total_custs>24989</total_custs>
      </area>
      <area>
        <area_name>Lee (SC)</area_name>
        <custs_out>0</custs_out>
        <total_custs>5517</total_custs>
      </area>
      <area>
        <area_name>Clarendon (SC)</area_name>
        <custs_out>0</custs_out>
        <total_custs>6036</total_custs>
      </area>
      <area>
        <area_name>Harnett (NC)</area_name>
        <custs_out>0</custs_out>
        <total_custs>33445</total_custs>
      </area>
    </areas>
  </curr_custs_aff>
</root>
```

# How to get the data

---

1. Scraping
2. Site's API (if it offers one)

API is application programming interface.

A site that offers an API has decided to make a standard way of sharing its data



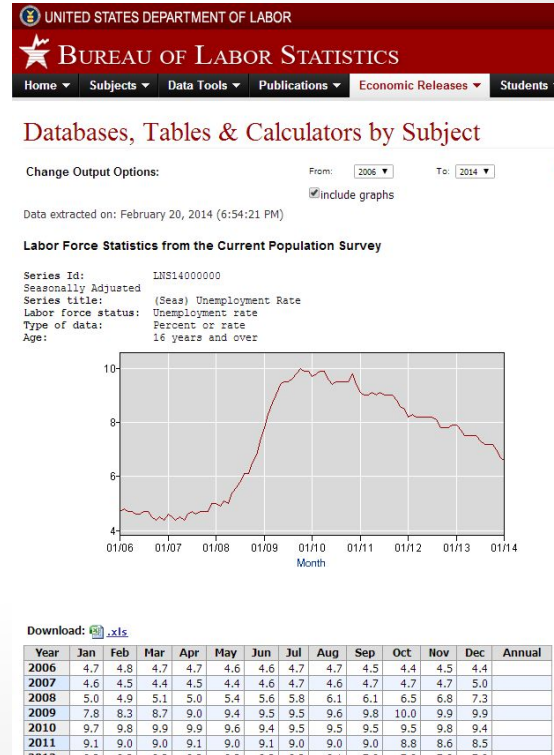
# Examples of data sources with APIs

---



# U.S. unemployment rate

How do you get a graph like this on your website and make it update automatically?



# The good news ...

---

## The U.S. BLS does offer an API

<b>HTTP Type:</b>	GET
<b>URL:</b>	<code>http://api.bls.gov/publicAPI/v1/timeseries/data/ &lt;series_id&gt;</code>
<b>Payload:</b>	<code>series_id</code>
<b>Example Payload:</b>	<code>LAUCN21219003</code>

Source: [http://www.bls.gov/developers/api\\_signature.htm](http://www.bls.gov/developers/api_signature.htm)

# The good news ... (con't.)



The screenshot shows a web browser window with the URL `api.bls.gov/publicAPI/v1/timeseries/data/LNS14000000`. A red box highlights the `LNS14000000` part of the URL, labeled "Series ID". The browser's developer tools show the JSON response for this endpoint:

```
{ "status": "REQUEST_SUCCEEDED", "responseTime": 12, "message": [], "Results": { "series": [ { "seriesID": "LNS14000000", "data": [ { "year": "2014", "period": "M01", "periodName": "January", "value": "6.6", "footnotes": [{}]}, { "year": "2013", "period": "M12", "periodName": "December", "value": "6.7", "footnotes": [{}]}, { "year": "2013", "period": "M11", "periodName": "November", "value": "7.0", "footnotes": [{}]}, { "year": "2013", "period": "M10", "periodName": "October", "value": "7.2", "footnotes": [{}]}, { "year": "2013", "period": "M09", "periodName": "September", "value": "7.2", "footnotes": [{}]}, { "year": "2013", "period": "M08", "periodName": "August", "value": "7.2", "footnotes": [{}]}, { "year": "2013", "period": "M07", "periodName": "July", "value": "7.3", "footnotes": [{}]}, { "year": "2013", "period": "M06", "periodName": "June", "value": "7.5", "footnotes": [{}]}, { "year": "2013", "period": "M05", "periodName": "May", "value": "7.5", "footnotes": [{}]}, { "year": "2013", "period": "M04", "periodName": "April", "value": "7.5", "footnotes": [{}]}, { "year": "2013", "period": "M03", "periodName": "March", "value": "7.5", "footnotes": [{}]}, { "year": "2013", "period": "M02", "periodName": "February", "value": "7.7", "footnotes": [{}]}, { "year": "2013", "period": "M01", "periodName": "January", "value": "7.9", "footnotes": [{}]}, { "year": "2012", "period": "M12", "periodName": "December", "value": "7.9", "footnotes": [{}]}, { "year": "2012", "period": "M11", "periodName": "November", "value": "7.8", "footnotes": [{}]}, { "year": "2012", "period": "M10", "periodName": "October", "value": "7.8", "footnotes": [{}]}, { "year": "2012", "period": "M09", "periodName": "September", "value": "7.8", "footnotes": [{}]}, { "year": "2012", "period": "M08", "periodName": "August", "value": "8.1", "footnotes": [{}]}, { "year": "2012", "period": "M07", "periodName": "July", "value": "8.2", "footnotes": [{}]}, { "year": "2012", "period": "M06", "periodName": "June", "value": "8.2", "footnotes": [{}]}, { "year": "2012", "period": "M05", "periodName": "May", "value": "8.2", "footnotes": [{}]}, { "year": "2012", "period": "M04", "periodName": "April", "value": "8.2", "footnotes": [{}]}, { "year": "2012", "period": "M03", "periodName": "March", "value": "8.2", "footnotes": [{}]}, { "year": "2012", "period": "M02", "periodName": "February", "value": "8.3", "footnotes": [{}]}, { "year": "2012", "period": "M01", "periodName": "January", "value": "8.2", "footnotes": [{}]} ] } ] }
```

# The good news ... (con't.)

---

```
{
  "status": "REQUEST_SUCCEEDED",
  "responseTime": 12,
  "message": [

  ],
  "Results": {
    "series": [
      {
        "seriesID": "LNS1400000",
        "data": [
          {
            "year": "2014",
            "period": "M01",
            "periodName": "January",
            "value": "6.6",
            "footnotes": [
              {
            }
          ]
        }
      },
      {
        "year": "2013",
```

JSON  
JavaScript Object Notation

# The bad news ...

---

## Browsers cannot directly read this type of file

- Same origin policy: browsers do not allow JavaScript code to load a plain JSON file that resides on a different domain
- Exceptions: JSONP with callback, Cross-origin resource sharing (CORS)

Source: [http://en.wikipedia.org/wiki/Same\\_origin\\_policy](http://en.wikipedia.org/wiki/Same_origin_policy),  
[http://en.wikipedia.org/wiki/Cross-origin\\_resource\\_sharing](http://en.wikipedia.org/wiki/Cross-origin_resource_sharing)

# Solutions ...

---

“Proxy” or a go-between:

- Google spreadsheets
- Yahoo Query Language
- Your own program on a server acting as a proxy

# Google spreadsheets

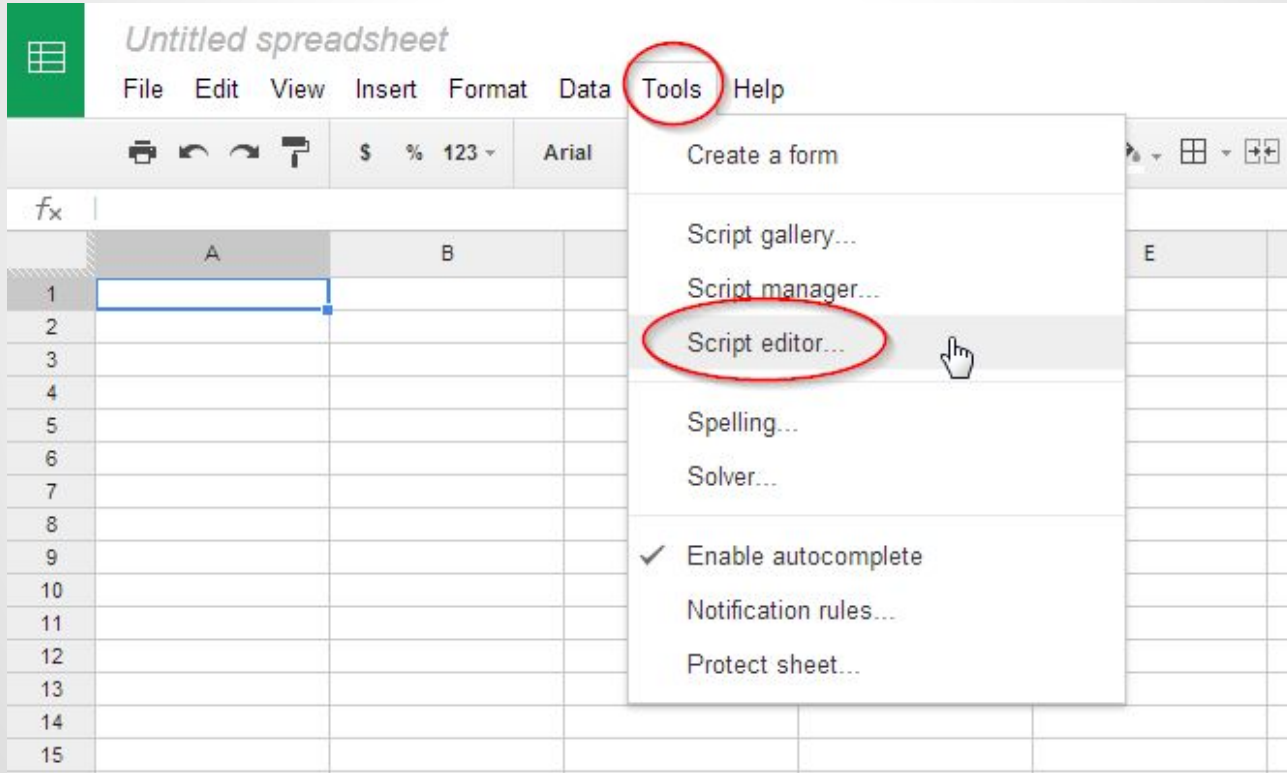
---

- Google Application Script can fetch data from an API and store it in a spreadsheet
- A script can be set up to run automatically according to a schedule
- Data can be pulled out of a spreadsheet and used for visualizations



# Google spreadsheets

---



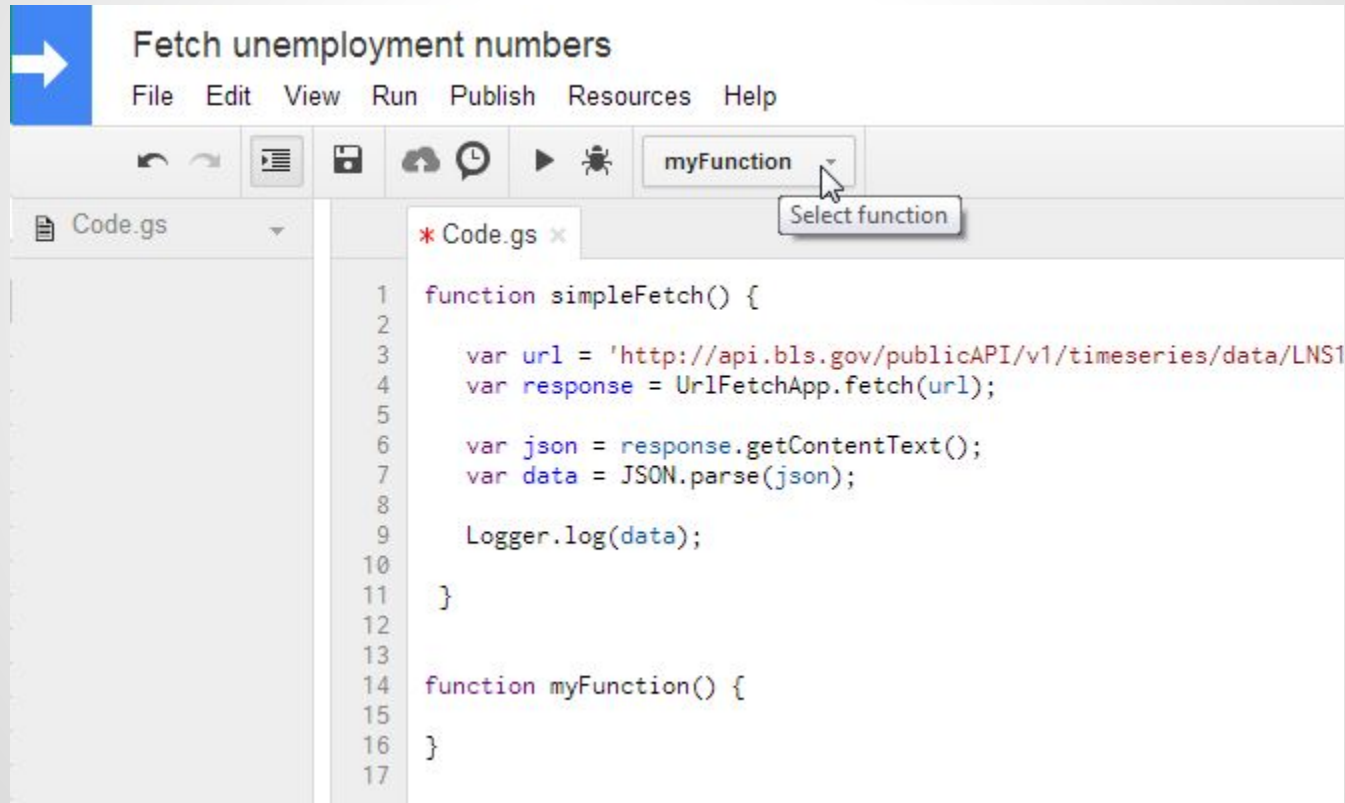
# Google spreadsheets

---

```
function simpleFetch() {  
  
    var url = 'http://api.bls.gov/publicAPI/v1/timeseries/data/LNS14000000';  
    var response = UrlFetchApp.fetch(url);  
  
    var json = response.getContentText();  
    var data = JSON.parse(json);  
  
    Logger.log(data);  
  
}
```

# Google spreadsheets

---



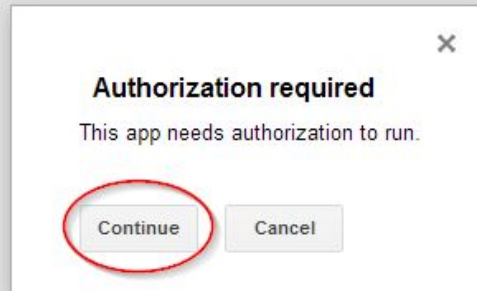
The screenshot shows the Google Apps Script editor interface. At the top, there is a blue header with a white arrow icon and the title "Fetch unemployment numbers". Below the header is a menu bar with "File", "Edit", "View", "Run", "Publish", "Resources", and "Help". A toolbar contains icons for undo, redo, list view, save, refresh, chat, play, and a bug icon. A dropdown menu labeled "myFunction" is open, showing a "Select function" option. The main editor area displays a code file named "Code.gs" with the following JavaScript code:

```
1 function simpleFetch() {  
2  
3   var url = 'http://api.bls.gov/publicAPI/v1/timeseries/data/LNS1  
4   var response = UrlFetchApp.fetch(url);  
5  
6   var json = response.getContentText();  
7   var data = JSON.parse(json);  
8  
9   Logger.log(data);  
10  
11 }  
12  
13  
14 function myFunction() {  
15  
16 }  
17
```

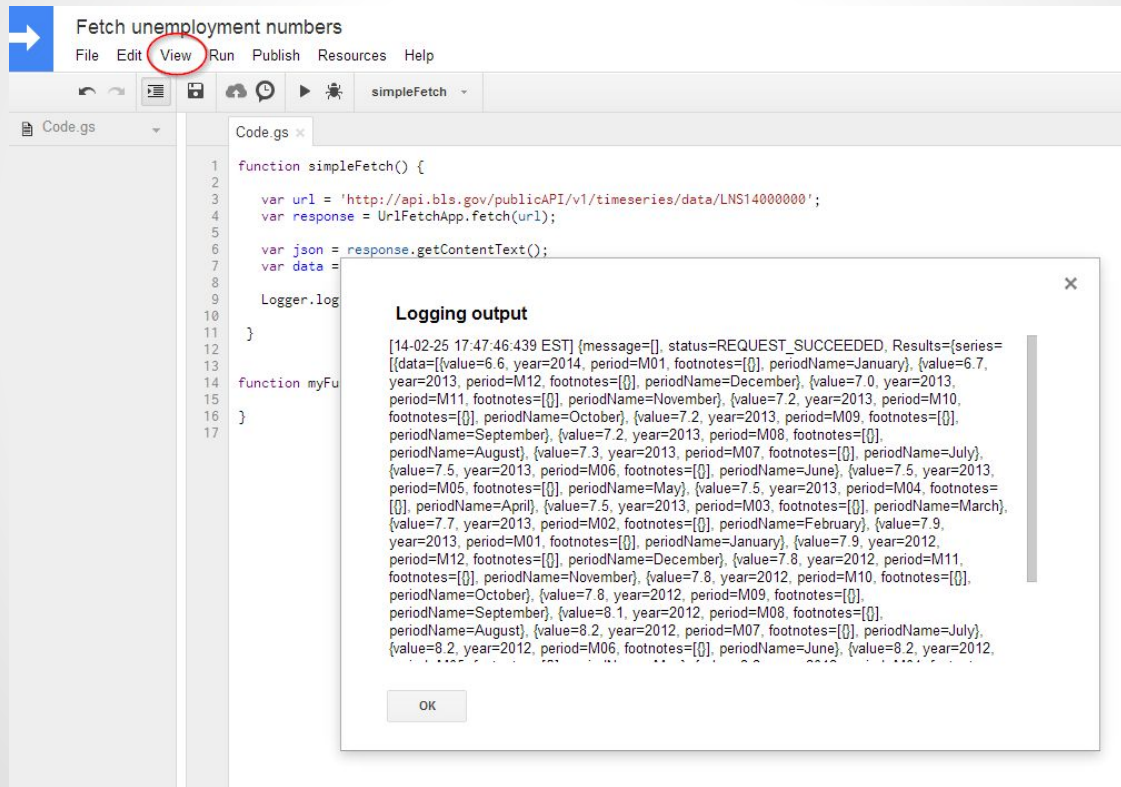
# Google spreadsheets

---

```
Code.gs x  
  
function simpleFetch() {  
  
  var url = 'http://api.bls.gov/publicAPI/v1/timeseries/data/LNS14000000';  
  var response = UrlFetchApp.fetch(url);  
  
  var json = response.getContentText();  
  var data = JSON.parse(json);  
  
  Logger.log(data);  
  
}  
  
function myFunction() {  
  
}
```



# Google spreadsheets



Fetch unemployment numbers

File Edit **View** Run Publish Resources Help

Code.gs

```
1 function simpleFetch() {
2
3   var url = 'http://api.bls.gov/publicAPI/v1/timeseries/data/LNS14000000';
4   var response = UrlFetchApp.fetch(url);
5
6   var json = response.getContentText();
7   var data =
8
9   Logger.log
10
11 }
12
13
14 function myFu
15
16 }
17
```

**Logging output**

[14-02-25 17:47:46:439 EST] {message=[], status=REQUEST\_SUCCEEDED, Results={series=[[data=[{value=6.6, year=2014, period=M01, footnotes=[], periodName=January}, {value=6.7, year=2013, period=M12, footnotes=[], periodName=December}, {value=7.0, year=2013, period=M11, footnotes=[], periodName=November}, {value=7.2, year=2013, period=M10, footnotes=[], periodName=October}, {value=7.2, year=2013, period=M09, footnotes=[], periodName=September}, {value=7.2, year=2013, period=M08, footnotes=[], periodName=August}, {value=7.3, year=2013, period=M07, footnotes=[], periodName=July}, {value=7.5, year=2013, period=M06, footnotes=[], periodName=June}, {value=7.5, year=2013, period=M05, footnotes=[], periodName=May}, {value=7.5, year=2013, period=M04, footnotes=[], periodName=April}, {value=7.5, year=2013, period=M03, footnotes=[], periodName=March}, {value=7.7, year=2013, period=M02, footnotes=[], periodName=February}, {value=7.9, year=2013, period=M01, footnotes=[], periodName=January}, {value=7.9, year=2012, period=M12, footnotes=[], periodName=December}, {value=7.8, year=2012, period=M11, footnotes=[], periodName=November}, {value=7.8, year=2012, period=M10, footnotes=[], periodName=October}, {value=7.8, year=2012, period=M09, footnotes=[], periodName=September}, {value=8.1, year=2012, period=M08, footnotes=[], periodName=August}, {value=8.2, year=2012, period=M07, footnotes=[], periodName=July}, {value=8.2, year=2012, period=M06, footnotes=[], periodName=June}, {value=8.2, year=2012, period=M05, footnotes=[], periodName=May}, {value=8.2, year=2012, period=M04, footnotes=[], periodName=April}, {value=8.2, year=2012, period=M03, footnotes=[], periodName=March}, {value=8.2, year=2012, period=M02, footnotes=[], periodName=February}, {value=8.2, year=2012, period=M01, footnotes=[], periodName=January}]]}}

OK

# Google spreadsheets

---

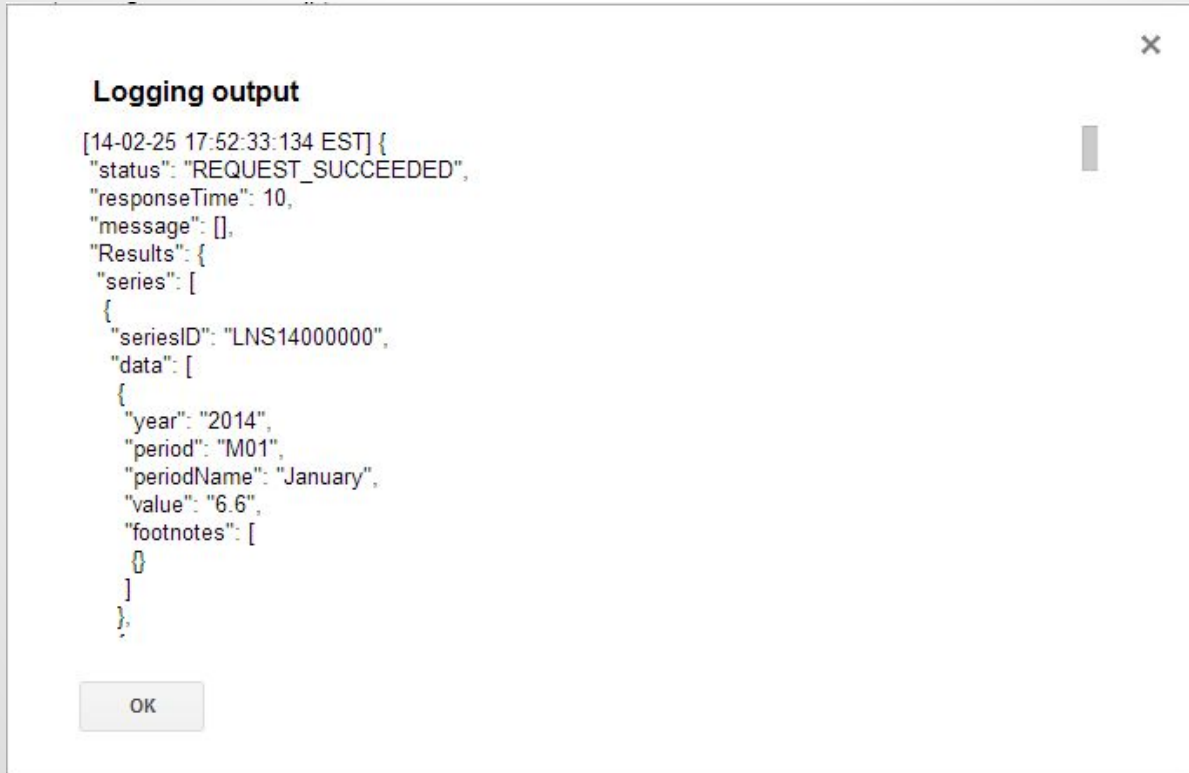
```
* Code.gs x
function simpleFetch() {
    var url = 'http://api.bls.gov/publicAPI/v1/timeseries/data/LNS14000000';
    var response = UrlFetchApp.fetch(url);

    var json = response.getContentText();
    var data = JSON.parse(json);
    |
    var niceData = JSON.stringify(data,null," ");
    Logger.log(niceData);
}

function myFunction() {
}
```

# Google spreadsheets

---

A dialog box titled "Logging output" with a close button (X) in the top right corner. It contains a JSON log entry. At the bottom left is an "OK" button.

```
[14-02-25 17:52:33:134 EST] {
  "status": "REQUEST_SUCCEEDED",
  "responseTime": 10,
  "message": [],
  "Results": {
    "series": [
      {
        "seriesID": "LNS14000000",
        "data": [
          {
            "year": "2014",
            "period": "M01",
            "periodName": "January",
            "value": "6.6",
            "footnotes": [
              {}
            ]
          }
        ]
      }
    ]
  }
}
```

OK

# Google spreadsheets

---

```
Logging output
[14-02-25 17:52:33:134 EST] {
  "status": "REQUEST_SUCCEEDED",
  "responseTime": 10,
  "message": [],
  "Results": {
    "series": [
      {
        "seriesID": "LNS14000000",
        "data": [
          {
            "year": "2014",
            "period": "M01",
            "periodName": "January",
            "value": "6.6",
            "footnotes": [
              {}
            ]
          }
        ]
      }
    ]
  }
}
```

OK



# Google spreadsheets

---

Code.gs x

```
function simpleFetch() {  
  
    var url = 'http://api.bls.gov/publicAPI/v1/timeseries/data/LNS14000000';  
    var response = UrlFetchApp.fetch(url);  
  
    var json = response.getContentText();  
    var data = JSON.parse(json);  
  
    var year = data.Results.series[0].data[0].year;  
    var month = data.Results.series[0].data[0].year;  
    var rate = data.Results.series[0].data[0].value;  
  
    Logger.log("Year: " + year + " Month: " + month + " Rate: " + rate + "%");  
  
}
```

# Google spreadsheets

---

## Logging output

[14-02-25 18:17:58:509 EST] Year: 2014 Month: 2014 Rate: 6.6%

# Google spreadsheets

---

```
function simpleFetch() {  
  
    var url = 'http://api.bls.gov/publicAPI/v1/timeseries/data/LNS14000000';  
    var response = UrlFetchApp.fetch(url);  
  
    var json = response.getContentText();  
    var data = JSON.parse(json);  
  
    var dataArray = data.Results.series[0].data;  
    var values = [];  
  
    for (var x=0; x<dataArray.length; x++) {  
        var thisMo = parseInt(dataArray[x].period.slice(-2),10);  
        var thisYr = dataArray[x].year;  
        var thisDy = daysInMonth(thisMo,thisYr); // last day of the month  
        var thisDateStr = thisMo + "/" + thisDy + "/" + thisYr;  
        var thisDate = new Date(thisDateStr);  
        values.push([thisDate,dataArray[x].value]);  
    }  
  
    // sort by date, ascending  
    values.sort(function(a,b){  
        return (a[0] - b[0]);  
    });  
  
    Logger.log(values);  
}
```

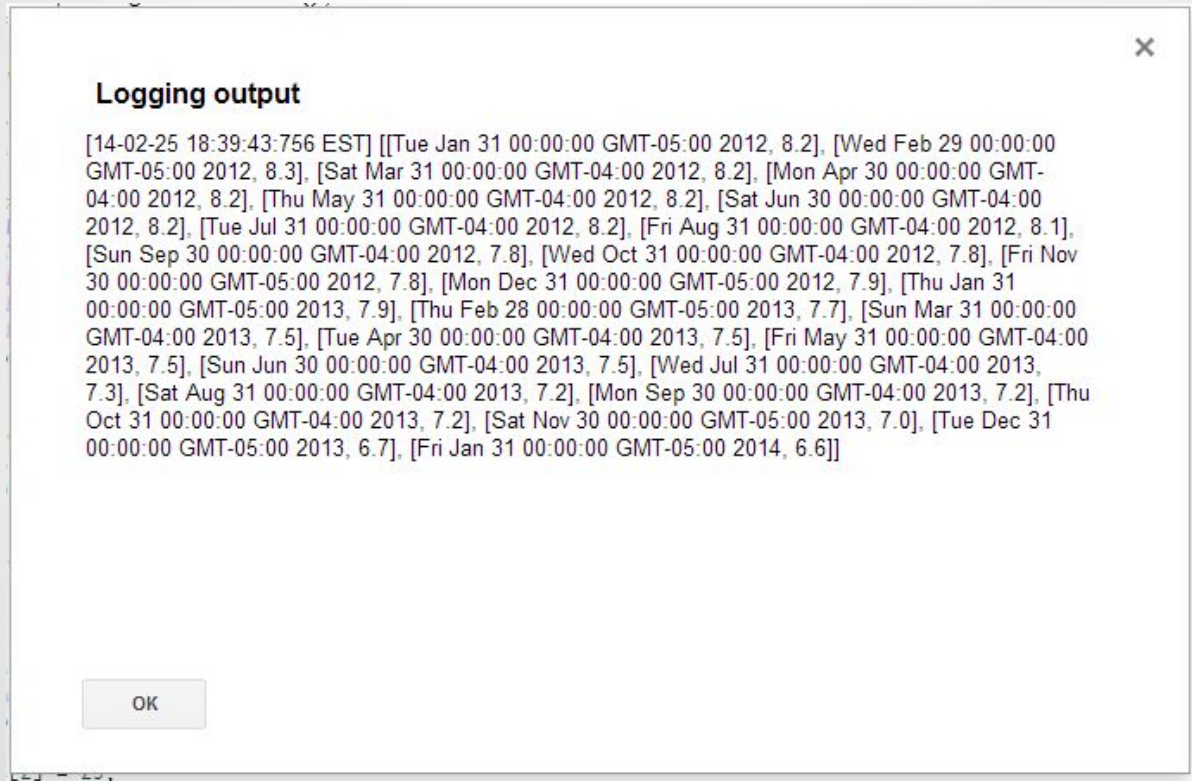
# Google spreadsheets

---

```
// -----  
function daysInMonth(mo,yr) {  
    var isLeap = ((yr % 4) == 0 && ((yr % 100) != 0 || (yr % 400) == 0));  
    var allMonths = [null,31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31];  
    if (isLeap) {  
        allMonths[2] = 29;  
    }  
    return allMonths[mo];  
}
```

# Google spreadsheets

---



# Google spreadsheets

---

\* Code.gs x

```
var ss = SpreadsheetApp.getActiveSpreadsheet();

if (ss.getSheetByName('us_unemployment')===null) {
  var sheet = ss.insertSheet('us_unemployment');
}
else {
  var sheet = ss.getSheetByName('us_unemployment');
}

var range = sheet.getDataRange();
var lastRow = range.getLastRow();

if (lastRow === 1) {
  range = sheet.getRange("A1:B1");
  range.setValues([[ 'Date', 'Rate' ]]);
}

var startRow = range.getLastRow() + 1;
range = sheet.getRange(startRow,1,values.length,values[0].length);
range.setValues(values);

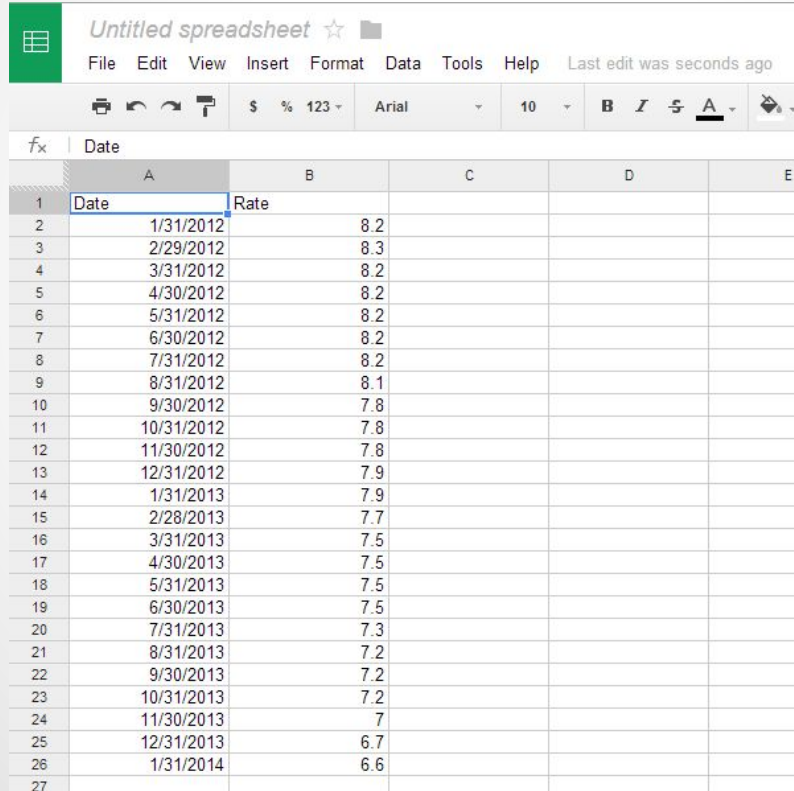
removeDuplicates();
}
```

# Google spreadsheets

---

```
// -----  
// from https://developers.google.com/apps-script/articles/removing\_duplicates  
function removeDuplicates() {  
  var sheet = SpreadsheetApp.getActiveSheet();  
  var data = sheet.getDataRange().getValues();  
  var newData = new Array();  
  for(i in data){  
    var row = data[i];  
    var duplicate = false;  
    for(j in newData){  
      if(row.join() == newData[j].join()){  
        duplicate = true;  
      }  
    }  
    if(!duplicate){  
      newData.push(row);  
    }  
  }  
  sheet.clearContents();  
  sheet.getRange(1, 1, newData.length, newData[0].length).setValues(newData);  
}
```

# Google spreadsheets

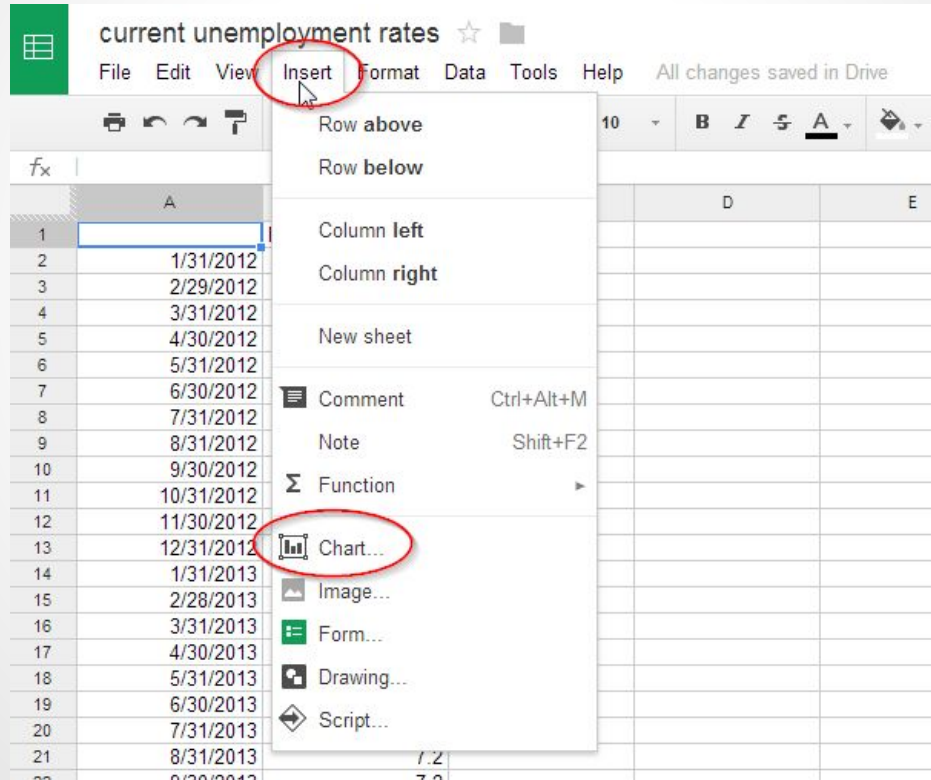


The screenshot shows a Google Spreadsheet titled "Untitled spreadsheet". The interface includes a menu bar (File, Edit, View, Insert, Format, Data, Tools, Help) and a toolbar with various icons for printing, undo, redo, and text formatting. The spreadsheet itself has a grid with columns A through E and rows 1 through 27. The first row (row 1) has a header "Date" in column A and "Rate" in column B. The subsequent rows contain a list of dates from 1/31/2012 to 1/31/2014 and corresponding numerical rates ranging from 8.2 down to 6.6.

	A	B	C	D	E
1	Date	Rate			
2	1/31/2012	8.2			
3	2/29/2012	8.3			
4	3/31/2012	8.2			
5	4/30/2012	8.2			
6	5/31/2012	8.2			
7	6/30/2012	8.2			
8	7/31/2012	8.2			
9	8/31/2012	8.1			
10	9/30/2012	7.8			
11	10/31/2012	7.8			
12	11/30/2012	7.8			
13	12/31/2012	7.9			
14	1/31/2013	7.9			
15	2/28/2013	7.7			
16	3/31/2013	7.5			
17	4/30/2013	7.5			
18	5/31/2013	7.5			
19	6/30/2013	7.5			
20	7/31/2013	7.3			
21	8/31/2013	7.2			
22	9/30/2013	7.2			
23	10/31/2013	7.2			
24	11/30/2013	7			
25	12/31/2013	6.7			
26	1/31/2014	6.6			
27					



# Google spreadsheets



The screenshot displays the Google Spreadsheets interface for a document titled "current unemployment rates". The menu bar includes File, Edit, View, Insert, Format, Data, Tools, and Help. The Insert menu is open, showing options such as Row above, Row below, Column left, Column right, New sheet, Comment, Note, Function, Chart..., Image..., Form..., Drawing..., and Script... The "Chart..." option is circled in red. The spreadsheet grid shows columns A through E and rows 1 through 22. The data in column A consists of dates from 1/31/2012 to 8/31/2013.

	A	D	E
1			
2	1/31/2012		
3	2/29/2012		
4	3/31/2012		
5	4/30/2012		
6	5/31/2012		
7	6/30/2012		
8	7/31/2012		
9	8/31/2012		
10	9/30/2012		
11	10/31/2012		
12	11/30/2012		
13	12/31/2012		
14	1/31/2013		
15	2/28/2013		
16	3/31/2013		
17	4/30/2013		
18	5/31/2013		
19	6/30/2013		
20	7/31/2013		
21	8/31/2013		
22	9/30/2013		

# Google spreadsheets

The screenshot displays the Google Spreadsheets interface. At the top, the title bar reads "current unemployment rates" with a star icon and a user profile "tbarn". Below this is a menu bar with "File", "Edit", "View", "Insert", "Format", "Data", "Tools", and "Help". A status bar indicates "Last edit was 5 minutes ago" and a "Comments" button is visible on the right. The main toolbar contains various icons for undo, redo, copy, paste, and other spreadsheet functions. The formula bar shows "fx Date".

The spreadsheet grid has columns A through I and rows 1 through 36. Row 1 contains the headers "Date" in column A and "Rate" in column B. Row 2 shows a date "1/31/2012" in column A and a rate "8.2" in column B. The "Chart Editor" dialog is open, showing a line chart titled "Chart title" with a legend for "Rate". The chart's horizontal axis is labeled "Horizontal axis title" and has major ticks for 4/1/2012, 10/1/2012, 4/1/2013, 10/1/2013, and 1/1/2014. The vertical axis ranges from 6.5 to 8.5. The "Customize" tab is active, showing options for title, legend, font, and background.

Date	Rate
1/31/2012	8.2
2/29/2012	
3/31/2012	
4/30/2012	
5/31/2012	
6/30/2012	
7/31/2012	
8/31/2012	
9/30/2012	
10/31/2012	
11/30/2012	
12/31/2012	
1/31/2013	
2/28/2013	
3/31/2013	
4/30/2013	
5/31/2013	
6/30/2013	
7/31/2013	
8/31/2013	
9/30/2013	
10/31/2013	
11/30/2013	
12/31/2013	
1/31/2014	

# Google spreadsheets



# Google spreadsheets

---

Publish chart ✕

Select a publish format:

Paste this into any HTML page:

```
<script type="text/javascript" src="//ajax.googleapis.com/ajax/static/modules/gviz/1.0/chart.js"> {"dataSourceUrl":"//docs.google.com/spreadsheet/tq?key=0Aqg9Vduc1JqEdE5MNzixdFVDdl9YRDh0OTIYMGhDb0E&transpose=0&headers=1&range=A1%3AB26&gid=1&pub=1","options":{"vAxes":[{"useFormatFromData":true,"title":null,"minValue":null,"viewWindow":{"min":null,"max":null},"maxValue":null},{"useFormatFromData":true,"minValue":null,"viewWindow":{"min":null,"max":null},"maxValue":null}],"titleTextStyle":{"bold":true,"color":"#000","fontSize":16},"booleanRole":"certainty","curveType":"function","title":"U.S. Unemployment","height":371,"animation":{"duration":500},"legend":"none","width":600,"lineWidth":2,"hAxis":{"useFormatFromData":true,"title":"","minValue":null,"viewWindowMode":null,"viewWindow":null,"maxValue":null},"state":{},"view":{},"isDefaultVisualization":false,"chartType":"LineChart","chartName":"Chart 1"}</script>
```

# Google spreadsheets

---

```
<html>
<head>
<style type="text/css">
</style>

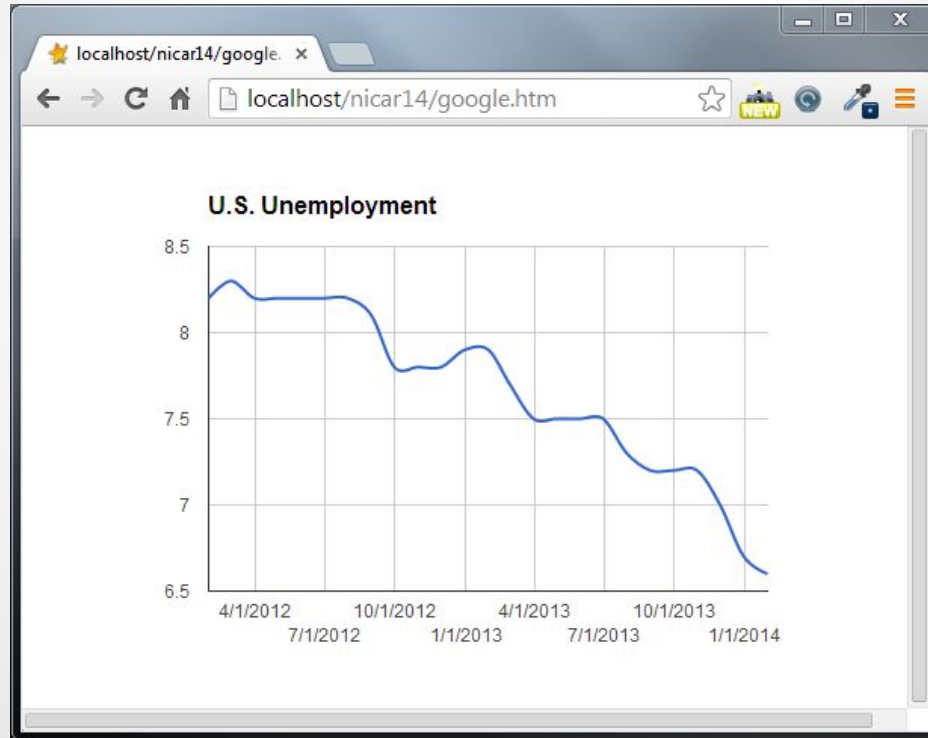
</head>
<body>

<script type="text/javascript" src="//ajax.googleapis.com/ajax/static/modules/gviz/1.0/chart.js">
{"dataSourceUrl":"//docs.google.com/spreadsheet/tq?key=0Aqg9Vduc1JqEdE5MNzkxdFVdd19YRDh0OT1YMGhDb0
E&transpose=0&headers=1&range=A1%3AB26&gid=1&pub=1", "options": {"vAxes": [{"useFormatFromData": true,
" title": null, "minValue": null, "viewWindow": {"min": null, "max": null}, "maxValue": null}, {"useFormatFrom
Data": true, "minValue": null, "viewWindow": {"min": null, "max": null}, "maxValue": null}], "titleTextStyle"
: {"bold": true, "color": "#000", "fontSize": 16}, "booleanRole": "certainty", "curveType": "function", "titl
e": "U.S.
Unemployment", "height": 371, "animation": {"duration": 500}, "legend": "none", "width": 600, "lineWidth": 2,
" hAxis": {"useFormatFromData": true, "title": "", "minValue": null, "viewWindowMode": null, "viewWindow": nu
ll, "maxValue": null}}, "state": {}, "view": {}, "isDefaultVisualization": false, "chartType": "LineChart", "
chartName": "Chart 1"} </script>

</body>
</html>
```

# Google spreadsheets

---



# Google spreadsheets

---

```
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.4.3/jquery.min.js"
type="text/javascript"></script>
<script type="text/javascript" src="https://www.google.com/jsapi"></script>
<script type="text/javascript">
google.load("visualization", '1', {packages:['corechart']});
google.setOnLoadCallback(drawChart);

function drawChart() {
  var url =
'https://docs.google.com/spreadsheet/pub?key=0Aqg9Vduc1JqEdE5MNzkxdFVDDl9YRDh00TlYMghDb0E&single=true&gid=1&output=html';
  var query = new google.visualization.Query(url);
  query.send(handleQueryResponse);
}
function handleQueryResponse(response) {
  if (response.isError()) {
    alert('Error in query: ' + response.getMessage() + ' ' + response.getDetailedMessage());
    return;
  }
}
// continued
```

# Google spreadsheets

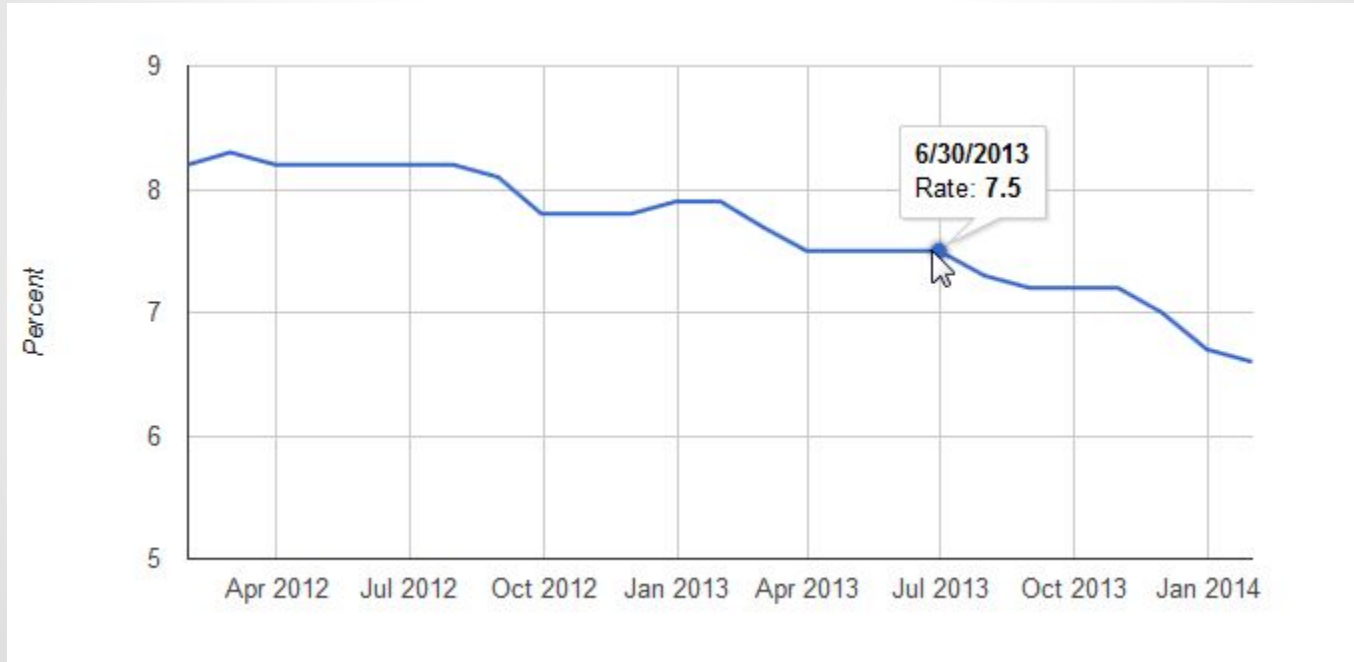
---

```
var data = response.getDataTable();
var chart = new google.visualization.LineChart(document.getElementById('columnchart'));
chart.draw(data,
  {
    legend: {
      position: 'none'
    },
    width:800,
    height:400,
    vAxis: {minValue: 5, title: 'Percent'}
  });
} // handleQueryResponse
</script>
<title>U.S. Unemployment rate</title>
</head>
<body>
<span id='columnchart' style="width:800px;"></span>
</body>
</html>
```



# Google spreadsheets

---



# Google spreadsheets

---

```
<html>
```

```
<head>
```

```
<link rel="stylesheet" type="text/css" href="js/jqplot/jquery.jqplot.css" />
```

```
<script src="js/jquery.min.js" type="text/javascript"></script>
```

```
<script type="text/javascript" src="js/jqplot/jquery.jqplot.js"></script>
```

```
<script type="text/javascript" src="js/jqplot/plugins/jqplot.highlighter.min.js"></script>
```

```
<script type="text/javascript" src="js/jqplot/plugins/jqplot.cursor.min.js"></script>
```

```
<script type="text/javascript"
```

```
src="js/jqplot/plugins/jqplot.dateAxisRenderer.min.js"></script>
```

```
<script type="text/javascript"
```

```
src="js/jqplot/plugins/jqplot.canvasTextRenderer.min.js"></script>
```

```
<script type="text/javascript"
```

```
src="js/jqplot/plugins/jqplot.canvasAxisLabelRenderer.min.js"></script>
```

# Google spreadsheets

---

```
<script type="text/javascript"
src='https://www.google.com/jsapi?autoload={"modules":[{"name":"visualization","version":"1
"}]}'></script>
<script type="text/javascript">

    google.setOnLoadCallback(drawVisualization);
    var line = [];

function drawVisualization() {
    var query = new google.visualization.Query(
'https://docs.google.com/spreadsheet/pub?key=0Aqg9Vduc1JqEdE5MNzkxDFVDdl9YRDh0OT1YMGhDb0E&s
heet=us_unemployment');
        query.send(handleQueryResponse);

    }
}
```

# Google spreadsheets

---

```
function handleQueryResponse(response) {  
  
    if (response.isError()) {  
        alert('Error in query: ' + response.getMessage() + ' ' +  
response.getDetailedMessage());  
        return;  
    }  
    $.each(response.n.xf, function() {  
        line.push([this.c[0].v, this.c[1].v]);  
    });  
  
    drawChart();  
  
}
```

# Google spreadsheets

---

```
function drawChart() {
  var plot1 = $.jqplot('chart1', [line], {
    title:'U.S. Unemployment Rate',
    width:960,
    series: [
      {
        lineWidth:3,
        markerOptions: {size:6 }
      }
    ],
    axesDefaults: {
      labelRenderer: $.jqplot.CanvasAxisLabelRenderer
    },
  },
```

# Google spreadsheets

---

```
axes:{
  xaxis:{
    renderer:$.jqplot.DateAxisRenderer,
    pad: 0,
    tickOptions:{
      formatString:'%b %Y'
    }
  },
  yaxis:{
    label: "Percent",
    tickOptions:{
      formatString:'%.1f'
    }
  }
},
```

# Google spreadsheets

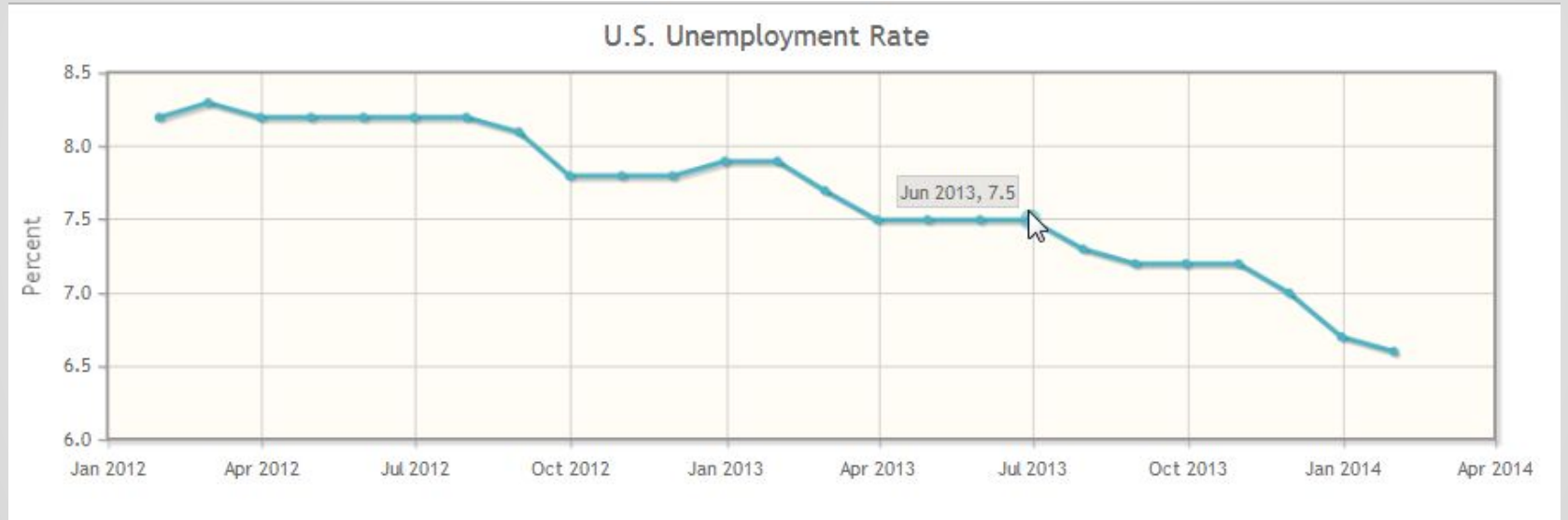
---

```
    highlighter: {
      show: true,
      sizeAdjust: 7.5
    },
    cursor: {
      show: false
    }
  });
}

</script>
</head>
<body>
<div id="chart1" width="960" style="width:960px;"></div>
</body>
</html>
```

# Google spreadsheets

---





# Google spreadsheets

**Release Calendar** SHARE ON:

**BY MONTH**

NOVEMBER 2013

DECEMBER 2013

JANUARY 2014

FEBRUARY 2014

MARCH 2014

APRIL 2014

MAY 2014

JUNE 2014

JULY 2014

AUGUST 2014

SEPTEMBER 2014

OCTOBER 2014

NOVEMBER 2014

DECEMBER 2014

ENTIRE YEAR, 2014

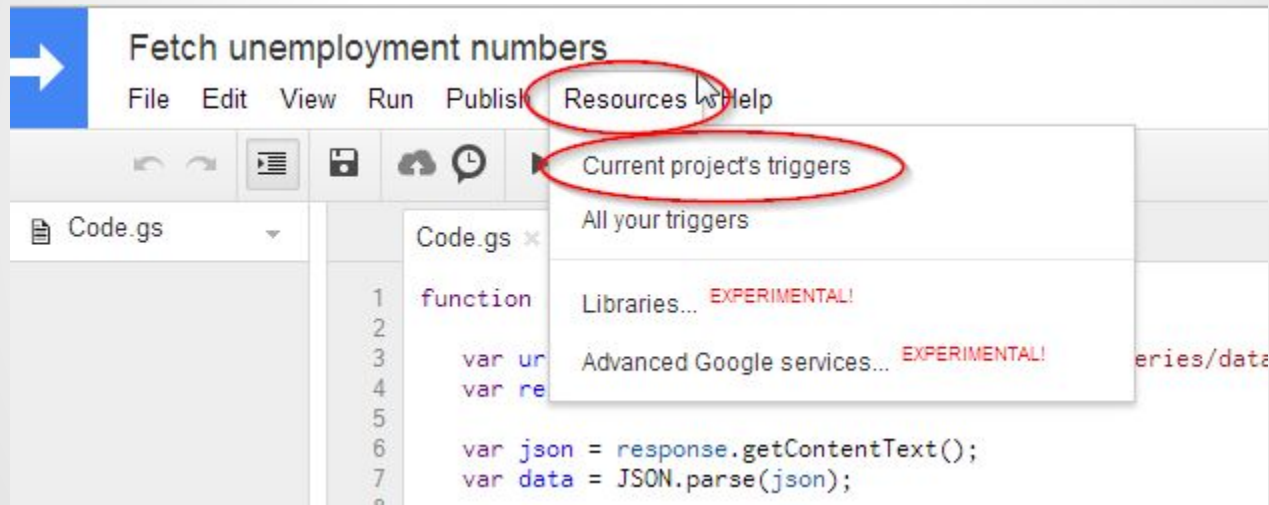
**Schedule of Releases for the Employment Situation**

Reference Month	Release Date	Release Time
October 2013	Nov. 08, 2013	08:30 AM
November 2013	Dec. 06, 2013	08:30 AM
December 2013	Jan. 10, 2014	08:30 AM
January 2014	Feb. 07, 2014	08:30 AM
February 2014	Mar. 07, 2014	08:30 AM
March 2014	Apr. 04, 2014	08:30 AM
April 2014	May 02, 2014	08:30 AM
May 2014	Jun. 06, 2014	08:30 AM
June 2014	Jul. 03, 2014	08:30 AM
July 2014	Aug. 01, 2014	08:30 AM
August 2014	Sep. 05, 2014	08:30 AM
September 2014	Oct. 03, 2014	08:30 AM
October 2014	Nov. 07, 2014	08:30 AM
November 2014	Dec. 05, 2014	08:30 AM

Source: [http://www.bls.gov/schedule/news\\_release/empsit.htm](http://www.bls.gov/schedule/news_release/empsit.htm)

# Google spreadsheets

---



# Google spreadsheets

---

✕

### Current project's triggers

Run	Events		
<span data-bbox="183 606 212 627">✕</span> simpleFetch	Time-driven	Day timer	9am to 10am
	(GMT-05:00) Eastern Time <a href="#">notifications</a>		

[Add a new trigger](#)

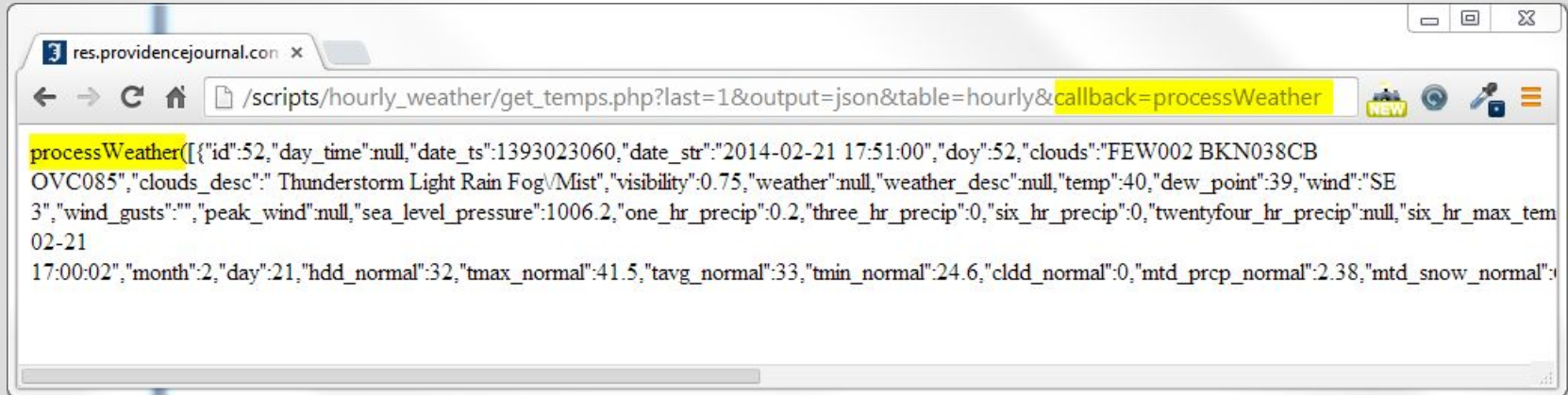
Save

Cancel

21 values.sort(function(a,b){

# What is a callback?

---



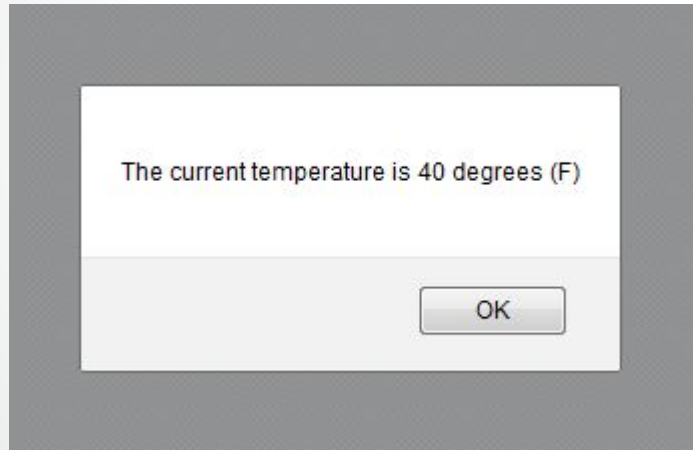
# Using callbacks with jQuery

---

```
<html>
<head>
<script src="http://code.jquery.com/jquery-1.10.1.min.js"></script>
<script type="text/javascript">
  function processWeather(data) {
    alert("The current temperature is " + data[0].temp + " degrees (F)");
  }
  $.ajax({
url:"http://res.providencejournal.com/weather/assets/scripts/hourly_weather/get_temps.php?last=1&output=json&table=hourly",
  type: "GET",
  dataType: "jsonp",
  success: processWeather,
  crossDomain: true
  });
</script>
</head>
<body>
</body>
</html>
```

# Result ...

---



# Using callback, jQuery to graph data

---

```
<html>
<head>
  <link rel="stylesheet" type="text/css"
href="http://res.providencejournal.com/global-assets/styles/jquery.jqplot.css" />
  <script type="text/javascript"
src="http://res.providencejournal.com/global-assets/scripts/jquery-1.8.3.js"></script>
  <script language="javascript" type="text/javascript"
src="http://res.providencejournal.com/global-assets/scripts/jquery.jqplot.min.js"></script>
  <script type="text/javascript"
src="http://res.providencejournal.com/global-assets/scripts/underscore-min.js"></script>
  <script language="javascript" type="text/javascript"
src="http://res.providencejournal.com/global-assets/scripts/jqplot.dateAxisRenderer.min.js"></script>
  <script type="text/javascript"
src="http://res.providencejournal.com/global-assets/scripts/jqplot.highlighter.js"></script>
  <script type="text/javascript">

    $(document).ready(function() {
```

# Using callback, jQuery to graph data

---

```
function processWeather(data){
  var times = _.map(_.pluck(data,'date_ts'),function(num){return num * 1000});
  var temperatures = _.pluck(data,'temp');
  var temperature_data = _.zip(times,temperatures);
  var ticks = _.map(_.pluck(data,'date_ts'),function(num){return (Math.round(num/3600)*3600)* 1000});
  var plot = $.jqplot('temperatures', [temperature_data], {
    title: 'Providence temperatures past 24 hours',
    axes:{
      xaxis:{
        renderer:$.jqplot.DateAxisRenderer,
        tickInterval: '4 hour',
        min:ticks[0],
        max:ticks[ticks.length-1],
        tickOptions:{ formatString:'%b %#d<br/>%#I:%M%p'}
      },
      yaxis:{
        tickOptions:{formatString:'%d&deg;F'}
      }
    },
    highlighter: {show: true,sizeAdjust: 7.5},
    cursor: {show: false}
  }); // plot
} // processWeather()
```



# Using callback, jQuery to graph data

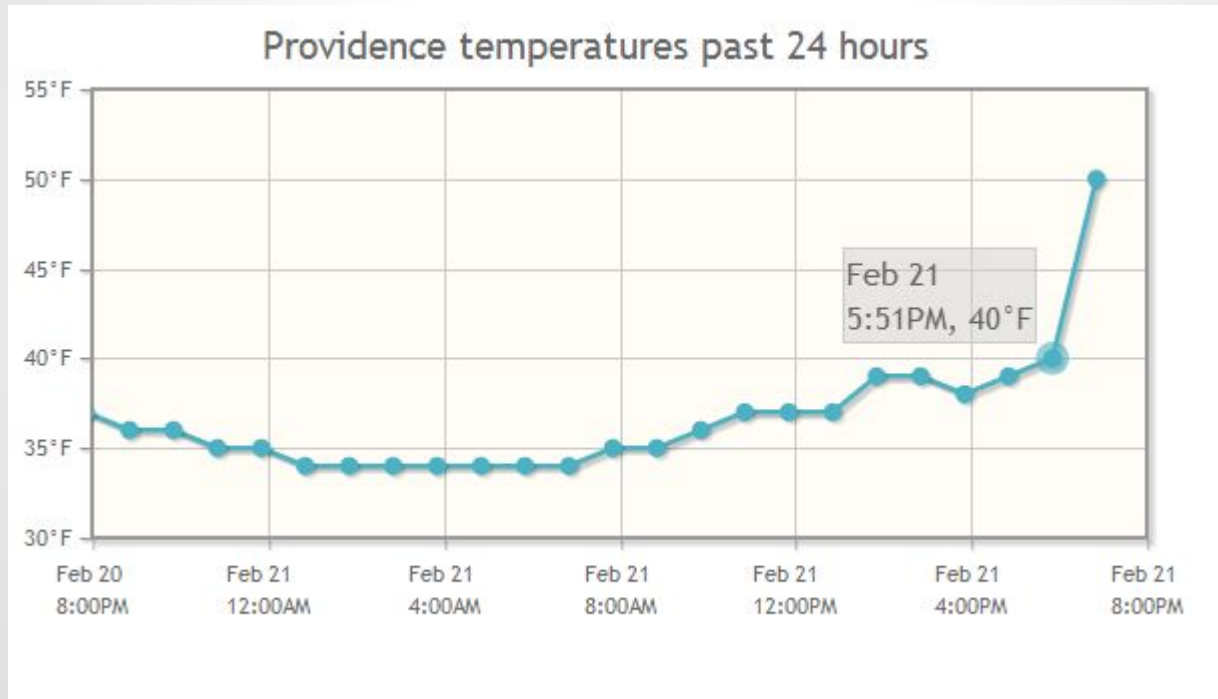
---

```
$.ajax({
  url:"http://res.providencejournal.com/weather/assets/scripts/hourly_weather/get_temps.php?last=24&ou
  tput=json&table=hourly",
  type: "GET",
  dataType:"jsonp",
  success: processWeather,
  crossDomain: true
}); // ajax
}); // document ready

</script>
</head>
<body>
  <div id="temperatures"></div>
</body>
</html>
```

# Result ...

---



# Questions ...

---