

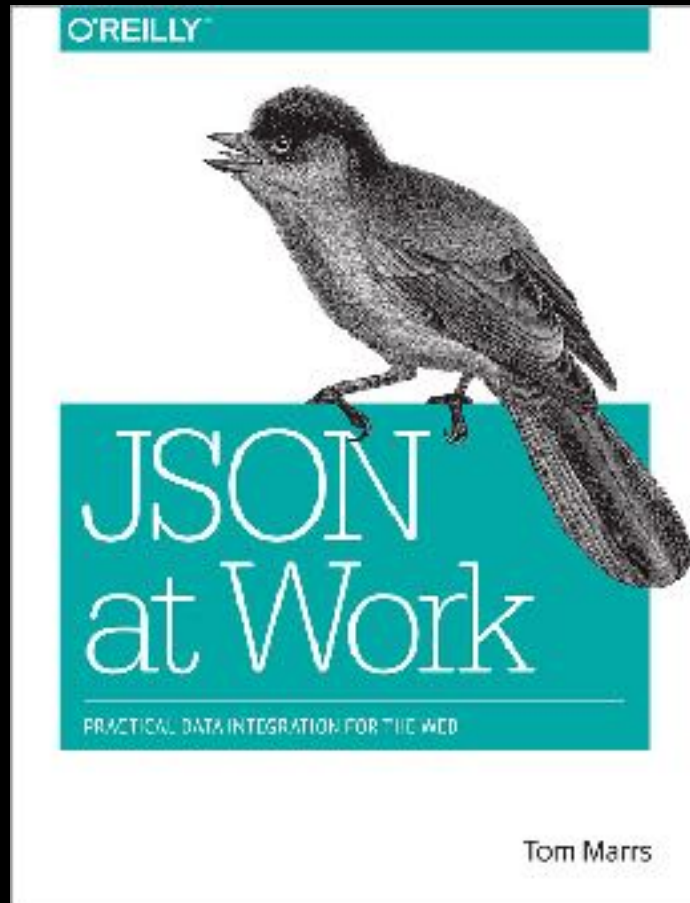
Bitter JSON

(Or When Complex Data Happens to Good Developers)

Tom Marrs



About Me ...



What's The Point?

JSON Search and Transform

Simplify interaction with RESTful APIs

Your Takeaway

jq + Handlebars Rock!

Agenda

JSON Search & Transform Overview	1
---	----------

JSON Search	2
--------------------	----------

JSON Transform	3
-----------------------	----------

We're Not Covering

REST

Deep JS

Other Languages

Examples and Slides

<https://github.com/tmarrs/presentations/tree/master/CS-OSUG/2017/Bitter-JSON>

Where Are We?

JSON Search & Transform Overview	1
---	----------

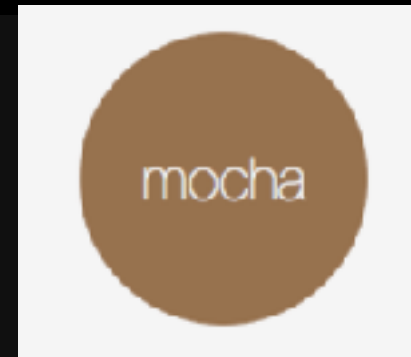
JSON Search	2
--------------------	----------

JSON Transform	3
-----------------------	----------

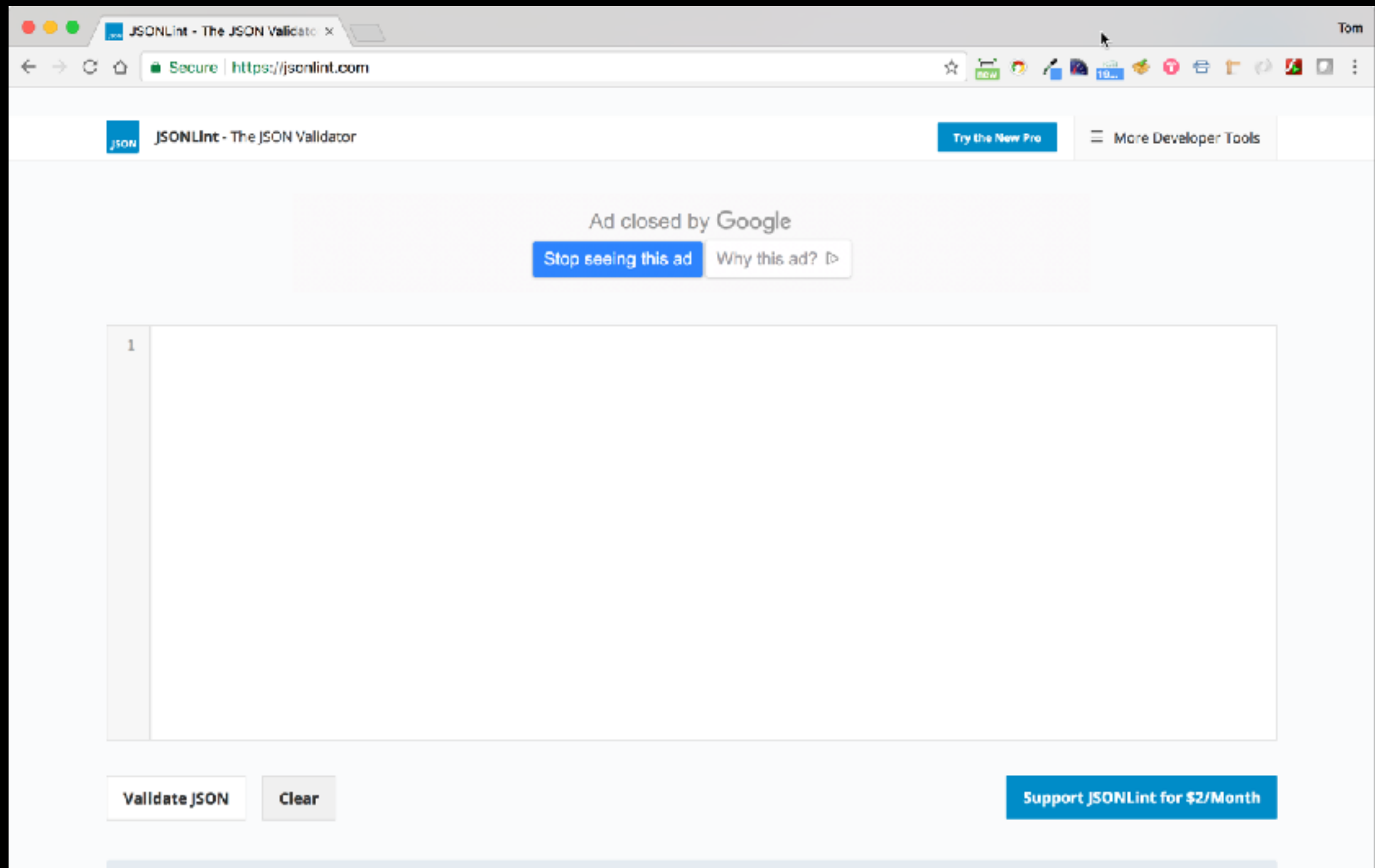
The Journey ...



Our Client Stack



JSONLint - Validation



JSON Search & Transform Rubric



Where Are We?

JSON Search & Transform Overview 1

JSON Search 2

JSON Transform 3

JSON Search Tools

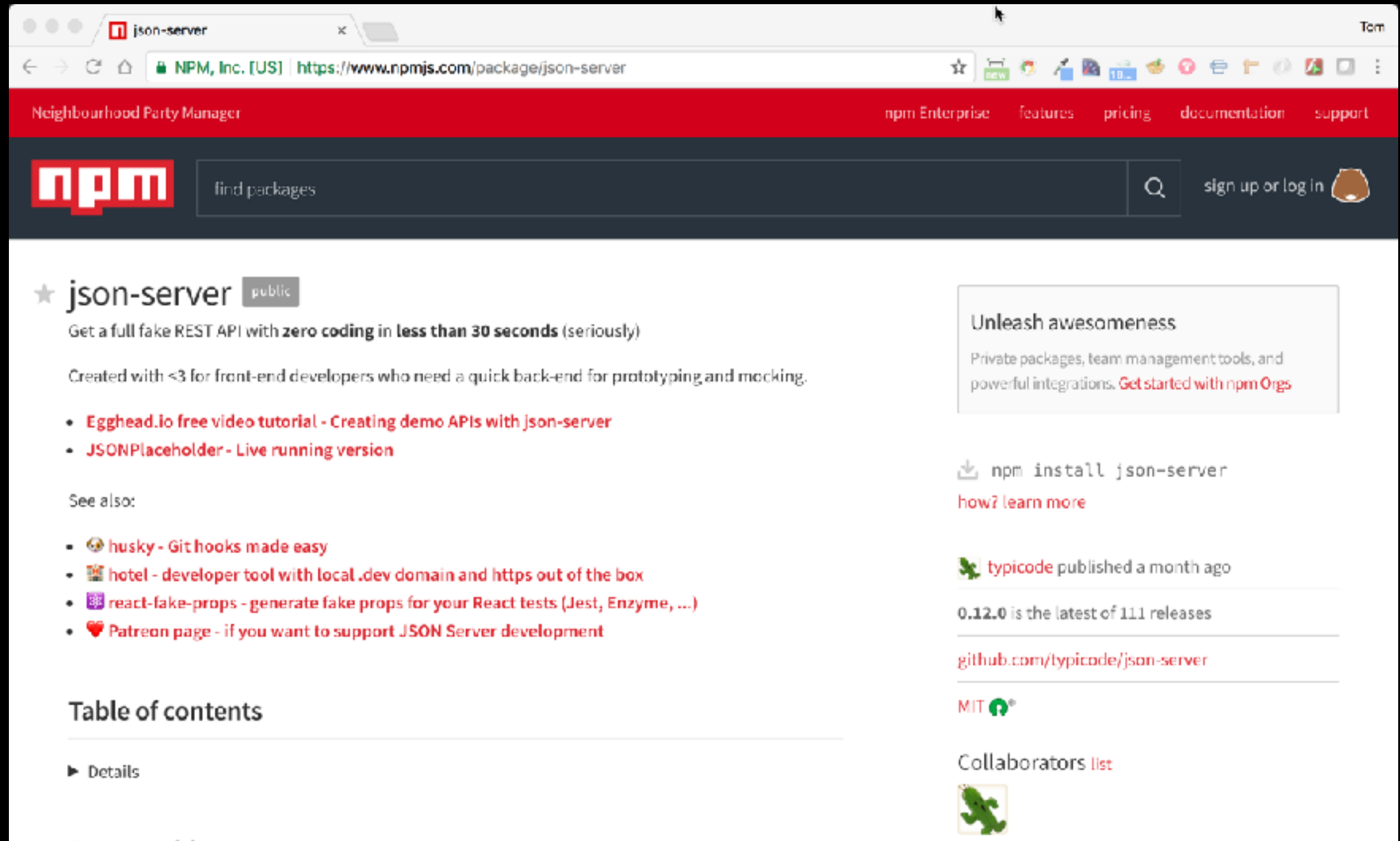


Weather Data API - CLI

```
json-server -p 5000 \  
cities-weather.json
```

<http://localhost:5000/cities>

json-server on npm



The screenshot shows the npm website interface for the 'json-server' package. The browser's address bar displays 'https://www.npmjs.com/package/json-server'. The page features a red navigation bar with links to 'npm Enterprise', 'features', 'pricing', 'documentation', and 'support'. Below this is a search bar with the text 'find packages' and a 'sign up or log in' button. The main content area for 'json-server' includes a star icon, the package name, and a 'public' label. The description states: 'Get a full fake REST API with zero coding in less than 30 seconds (seriously)'. It also mentions 'Created with <3 for front-end developers who need a quick back-end for prototyping and mocking.' A list of links includes 'Egghead.io free video tutorial - Creating demo APIs with json-server' and 'JSONPlaceholder - Live running version'. A 'See also:' section lists related packages like 'husky', 'hotel', 'react-fake-props', and a 'Patreon page'. On the right, a box titled 'Unleash awesomeness' promotes 'npm Orgs'. Below this, the command 'npm install json-server' is shown, followed by 'how? learn more'. It also notes that 'typicode published a month ago' and that '0.12.0 is the latest of 111 releases'. The GitHub repository link 'github.com/typicode/json-server' is provided, along with the MIT license icon. At the bottom, a 'Collaborators list' is shown with a single collaborator's profile picture.

json-server

NPM, Inc. [US] <https://www.npmjs.com/package/json-server>

Neighbourhood Party Manager

npm Enterprise features pricing documentation support

npm find packages

sign up or log in

★ json-server public

Get a full fake REST API with **zero coding in less than 30 seconds** (seriously)

Created with <3 for front-end developers who need a quick back-end for prototyping and mocking.

- [Egghead.io free video tutorial - Creating demo APIs with json-server](#)
- [JSONPlaceholder - Live running version](#)

See also:

- [husky - Git hooks made easy](#)
- [hotel - developer tool with local .dev domain and https out of the box](#)
- [react-fake-props - generate fake props for your React tests \(Jest, Enzyme, ...\)](#)
- [Patreon page - if you want to support JSON Server development](#)

Table of contents

- ▶ Details

Unleash awesomeness

Private packages, team management tools, and powerful integrations. [Get started with npm Orgs](#)

[npm install json-server](#)

[how? learn more](#)


[typicode](#) published a month ago

0.12.0 is the latest of 111 releases

github.com/typicode/json-server

MIT

Collaborators [list](#)



OpenWeather API - Postman

The screenshot displays the Postman application interface. The top bar includes tabs for Runner, Import, Builder, and Team Library. The left sidebar shows a search filter, a history of requests, and a collection of requests. The main area shows a GET request to `http://localhost:5000/cities`. The request is configured with no authorization and no body. The response is displayed in the bottom panel, showing a JSON array of city data. The status is 200 OK and the time taken is 66 ms.

Request URL: `http://localhost:5000/cities`

Method: GET

Authorization: No Auth

Body: (Empty)

Response Status: 200 OK, Time: 66 ms

Response Body (JSON):

```
[
  {
    "id": 5386095,
    "name": "Rancho Palos Verdes",
    "coord": {
      "lon": -118.387816,
      "lat": 33.744461
    },
    "main": {
      "temp": 84.34,
      "pressure": 1012,
      "humidity": 58,
      "temp_min": 78.8,
      "temp_max": 93
    },
    "dt": 1442171078,
    "wind": {
      "speed": 4.1,
      "deg": 300
    },
    "clouds": {
      "all": 5
    },
    "weather": [
      {
        "id": 800,
        "main": "Clear",
        "description": "Sky is Clear",
        "icon": "02d"
      }
    ]
  },
  {
    "id": 5392528,
    "name": "San Pedro"
  }
]
```

JSONPath

The screenshot shows a web browser window with the address bar displaying `goessner.net/articles/JsonPath/`. The page header includes the author's name `<stefan.goessner/>` and a navigation menu with links: `| Home | Lehre | Download | Info |`. The article title is `# JSONPath - XPath for JSON`, dated `| 2007-02-21 |`. The main text discusses the advantages of XML and introduces JSONPath as a tool for extracting data from JSON structures. It includes a list of bullet points and examples of XPath expressions for JSON, such as `/store/book[1]/title` and `x['store']['book'][0]['title']`. The right sidebar contains a search box, a table of contents under `>> Inhalt ..`, and a comments section at the bottom.

`<stefan.goessner/>`
Mechanik, das Web und der ganze Rest
[Home](#) | [Lehre](#) | [Download](#) | [Info](#)

JSONPath - XPath for JSON | 2007-02-21 | 1

A frequently emphasized advantage of XML is the availability of plenty tools to analyse, transform and selectively extract data out of XML documents. [XPath](#) is one of these powerful tools.

It's time to wonder, if there is a need for something like XPath4JSON and what are the problems it can solve.

- Data may be interactively found and extracted out of [JSON](#) structures on the client without special scripting.
- JSON data requested by the client can be reduced to the relevant parts on the server, such minimizing the bandwidth usage of the server response.

If we agree, that a tool for picking parts out of a JSON structure at hand does make sense, some questions come up. How should it do its job? How do JSONPath expressions look like?

Due to the fact, that JSON is a natural representation of data for the C family of programming languages, the chances are high, that the particular language has native syntax elements to access a JSON structure.

The following XPath expression

```
/store/book[1]/title
```

would look like

```
x.store.book[0].title
```

or

```
x['store']['book'][0]['title']
```

In Javascript, Python and PHP with a variable `x` holding the JSON structure. Here we

>> Search ..

Web goessner.net
Google Search

>> Inhalt ..

- Home
- Lehre
 - Dynamik
- Articles
 - DOM Events
 - Wiky
 - 2D Vectors
 - Slideous
 - JsonT
 - JSONPath
 - SVG
- Download
- Admin
- Info

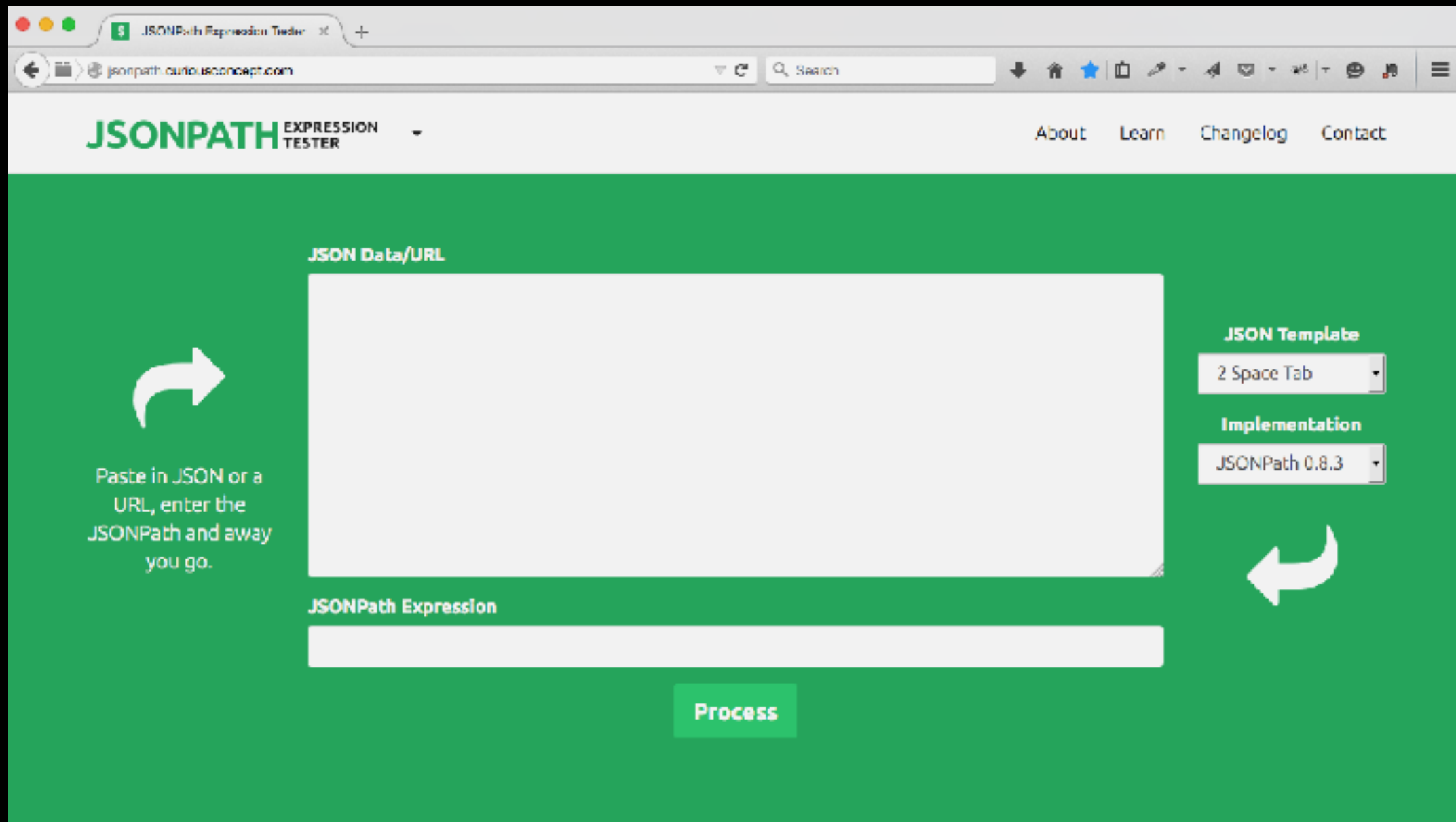
>> comments ..

Exercise 09

JSONPath Syntax

XPath	JSONPath	Result
/store/book/author	\$.store.book[*].author	the authors of all books in the store
//author	\$..author	all authors
/store/*	\$.store.*	all things in store, which are some books and a red bicycle.
/store//price	\$.store..price	the price of everything in the store.
//book[3]	\$..book[2]	the third book
//book[last()]	\$..book[(@.length-1)] \$..book[-1:]	the last book in order.
//book[position()<3]	\$..book[0,1] \$..book[:2]	the first two books
//book[isbn]	\$..book[?(@.isbn)]	filter all books with isbn number
//book[price<10]	\$..book[?(@.price<10)]	filter all books cheaper than 10
//*	\$..*	all Elements in XML document. All members of JSON structure.

JSONPath Expression Tester



The screenshot shows a web browser window with the title "JSONPath Expression Tester" and the URL "jsonpath.curiousconcept.com". The page has a green header with the logo "JSONPATH EXPRESSION TESTER" and navigation links "About", "Learn", "Changelog", and "Contact". The main content area has a green background. On the left, a white curved arrow points to a large white text area labeled "JSON Data/URL". Below this area is a smaller white text area labeled "JSONPath Expression". To the right of the "JSON Data/URL" area, there are two dropdown menus: "JSON Template" with "2 Space Tab" selected, and "Implementation" with "JSONPath 0.8.3" selected. A white curved arrow points from these dropdowns towards the "JSONPath Expression" area. At the bottom center, there is a green button labeled "Process".

JSONPATH EXPRESSION TESTER

About Learn Changelog Contact

JSON Data/URL

Paste in JSON or a URL, enter the JSONPath and away you go.

JSONPath Expression

JSON Template
2 Space Tab

Implementation
JSONPath 0.8.3

Process

JSONPath Online Evaluator

jsonpath online evaluator

jsonpath.com/?

Inputs

☐ Output paths

JSONPath Syntax

Example '\$.phoneNumbers[*].type' See also [JSONPath expressions](#)

JSON

```
3389     "name": "Moreno Valley",
3390     "coord": {
3391       "lon": -117.230591,
3392       "lat": 33.937519
3393     },
3394     "main": {
3395       "temp": 87.84,
3396       "pressure": 1013,
3397       "humidity": 42,
3398       "temp_min": 82.4,
3399       "temp_max": 98.6
3400     },
3401     "dt": 1442171075,
3402     "wind": {
3403       "speed": 1,
3404       "deg": 0
3405     },
3406     "clouds": {
3407       "all": 1
3408     },
3409     "weather": [
3410       {
3411         "id": 800,
3412         "main": "Clear",
3413         "description": "Sky is Clear",
3414         "icon": "01d"
3415       }
```

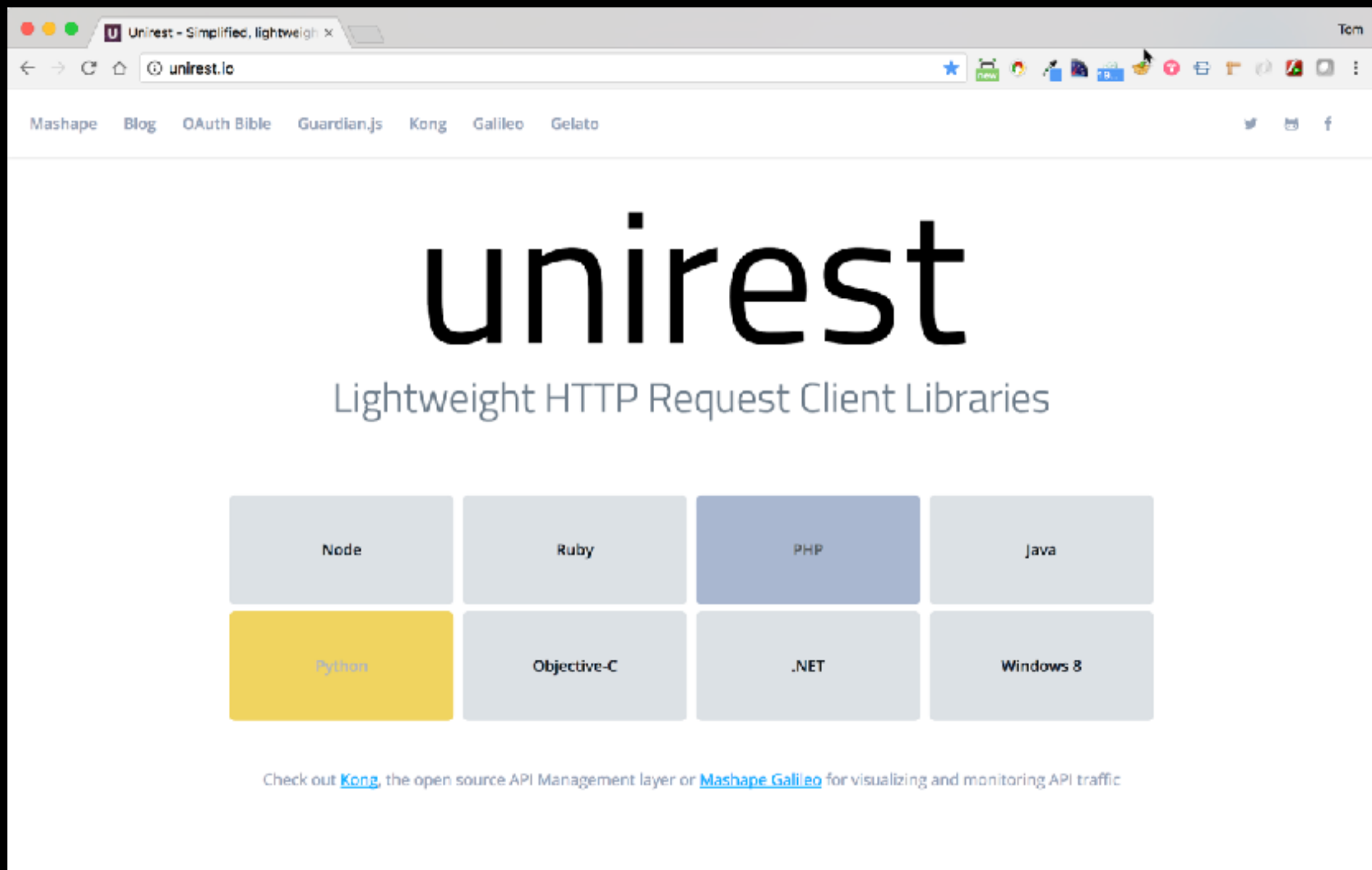
Evaluation Results

```
34     "id": 5392528,
35     "name": "San Pedro",
36     "coord": {
37       "lon": -118.29229,
38       "lat": 33.735851
39     },
40     "main": {
41       "temp": 84.02,
42       "pressure": 1012,
43       "humidity": 58,
44       "temp_min": 78.8,
45       "temp_max": 91
46     },
47     "dt": 1442171080,
48     "wind": {
49       "speed": 4.1,
50       "deg": 300
51     },
52     "clouds": {
53       "all": 5
54     },
55     "weather": [
56       {
57         "id": 800,
58         "main": "Clear",
59         "description": "Sky is Clear",
60         "icon": "02d"
61       }
62     ]
63   }
64 ]
```

JSONPath Queries

1	JSONPath query	Description
2	-----	
3	<code>\$.cities</code>	Get all elements in the cities Array.
4	<code>\$.cities.length</code>	Get the number of elements in the cities Array.
5	<code>\$.cities[0::2]</code>	Get every other element in the cities array.
6		
7	<code>\$.cities[(@.length-1)]</code>	Get the last element in the cities Array.
8		
9	<code>\$.weather</code>	Get all weather subelements.
10	<code>\$.cities[:3]</code>	Get the first three elements in cities Array.
11	<code>\$.cities[:3].name</code>	Get the city name for first three elements in the cities Array.
12	<code>\$.cities[?(@.main.temp > 84)]</code>	Get the cities where the temp > 84.
13		
14	<code>\$.cities[?(@.main.temp >= 84 && @.main.temp <= 85.5)]</code>	
15		Get the cities where the temp is between 84 and 85.5.
16		
17	<code>\$.cities[?(@.weather[0].main == 'Clouds')]</code>	
18		Get the cities with cloudy weather.
19		
20	<code>\$.cities[?(@.weather[0].main.match(/Clo/))]</code>	
21		Get the cities with cloudy weather by using regex.
22		

Unirest



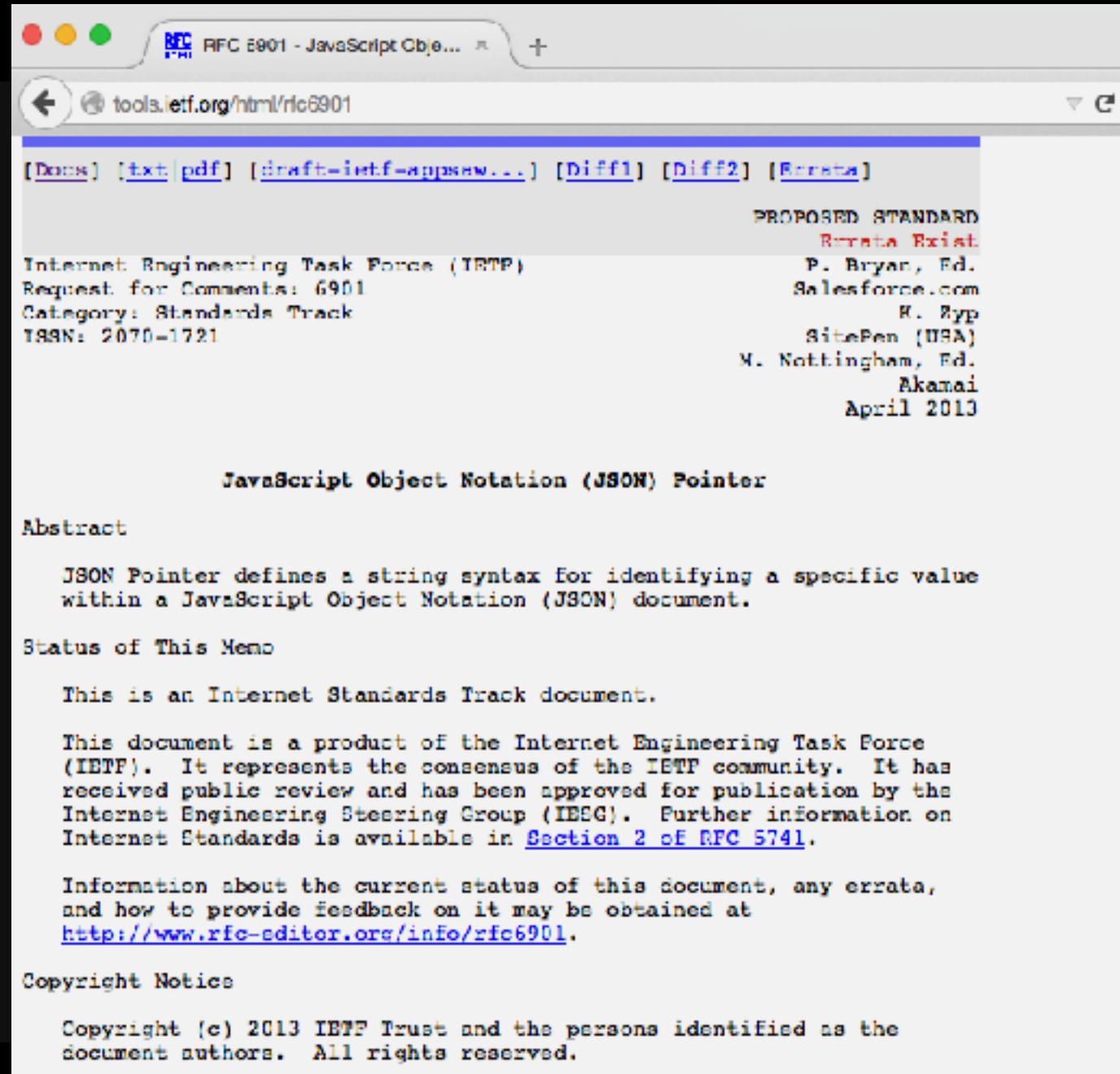
JSONPath Test

```
12  var expect = require('chai').expect;
13  var jp = require('jsonpath');
14  var unirest = require('unirest');
15
16  describe('cities-jsonpath', function() {
17    var req;
18
19    beforeEach(function() {
20      req = unirest.get('http://localhost:5000/cities')
21        .header('Accept', 'application/json');
22    });
23
24    it('should return a 200 response', function(done) {
25      req.end(function(res) {
26        expect(res.statusCode).to.eql(200);
27        expect(res.headers['content-type']).to.eql(
28          'application/json; charset=utf-8');
29        done();
30      });
31    });
32
33    it('should return all cities', function(done) {
34      req.end(function(res) {
35        var cities = res.body;
36
37        //console.log(cities);
38        expect(cities.length).to.eql(110);
39        done();
40      });
41    });
42  });
```


JSONPath Scorecard

Mindshare	Y
Dev Community	Y
Platforms	JS, Node.js, Java, RoR
Intuitive	Y
Standard	N

JSON Pointer



The screenshot shows a web browser window with the address bar displaying `tools.ietf.org/html/rfc6901`. The page content includes navigation links at the top: `[Doms]`, `[txt]`, `[pdf]`, `[draft-ietf-appsawg...`, `[Diff1]`, `[Diff2]`, and `[Errata]`. Below these links, the document is identified as a **PROPOSED STANDARD** with the status **Errata Exist**. The authors listed are P. Bryan, Ed. (Salesforce.com), H. Zyp (SitePen (USA)), and M. Nottingham, Ed. (Akamai), dated April 2013. The document title is **JavaScript Object Notation (JSON) Pointer**. The **Abstract** states: "JSON Pointer defines a string syntax for identifying a specific value within a JavaScript Object Notation (JSON) document." The **Status of This Memo** section notes it is an Internet Standards Track document and a product of the Internet Engineering Task Force (IETF). The **Copyright Notice** at the bottom states: "Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved."

[\[Doms\]](#) [\[txt\]](#) [\[pdf\]](#) [\[draft-ietf-appsawg...](#) [\[Diff1\]](#) [\[Diff2\]](#) [\[Errata\]](#)

PROPOSED STANDARD
Errata Exist

Internet Engineering Task Force (IETF)
Request for Comments: 6901
Category: Standards Track
ISSN: 2070-1721

P. Bryan, Ed.
Salesforce.com
H. Zyp
SitePen (USA)
M. Nottingham, Ed.
Akamai
April 2013

JavaScript Object Notation (JSON) Pointer

Abstract

JSON Pointer defines a string syntax for identifying a specific value within a JavaScript Object Notation (JSON) document.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in [Section 2 of RFC 5741](#).

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at <http://www.rfc-editor.org/info/rfc6901>.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

JSON Pointer Syntax

RFC 6901

JSON Pointer

April 2013

For example, given the JSON document

```
{
  "foo": ["bar", "baz"],
  "": 0,
  "a/b": 1,
  "c%d": 2,
  "e^f": 3,
  "g|h": 4,
  "i\\j": 5,
  "k\"l": 6,
  " ": 7,
  "m~n": 8
}
```

The following JSON strings evaluate to the accompanying values:

"	// the whole document
"/foo"	["bar", "baz"]
"/foo/0"	"bar"
"/"	0
"/a~1b"	1
"/c%d"	2
"/e^f"	3
"/g h"	4
"/i\\j"	5
"/k\"l"	6
"/ "	7
"/m~0n"	8

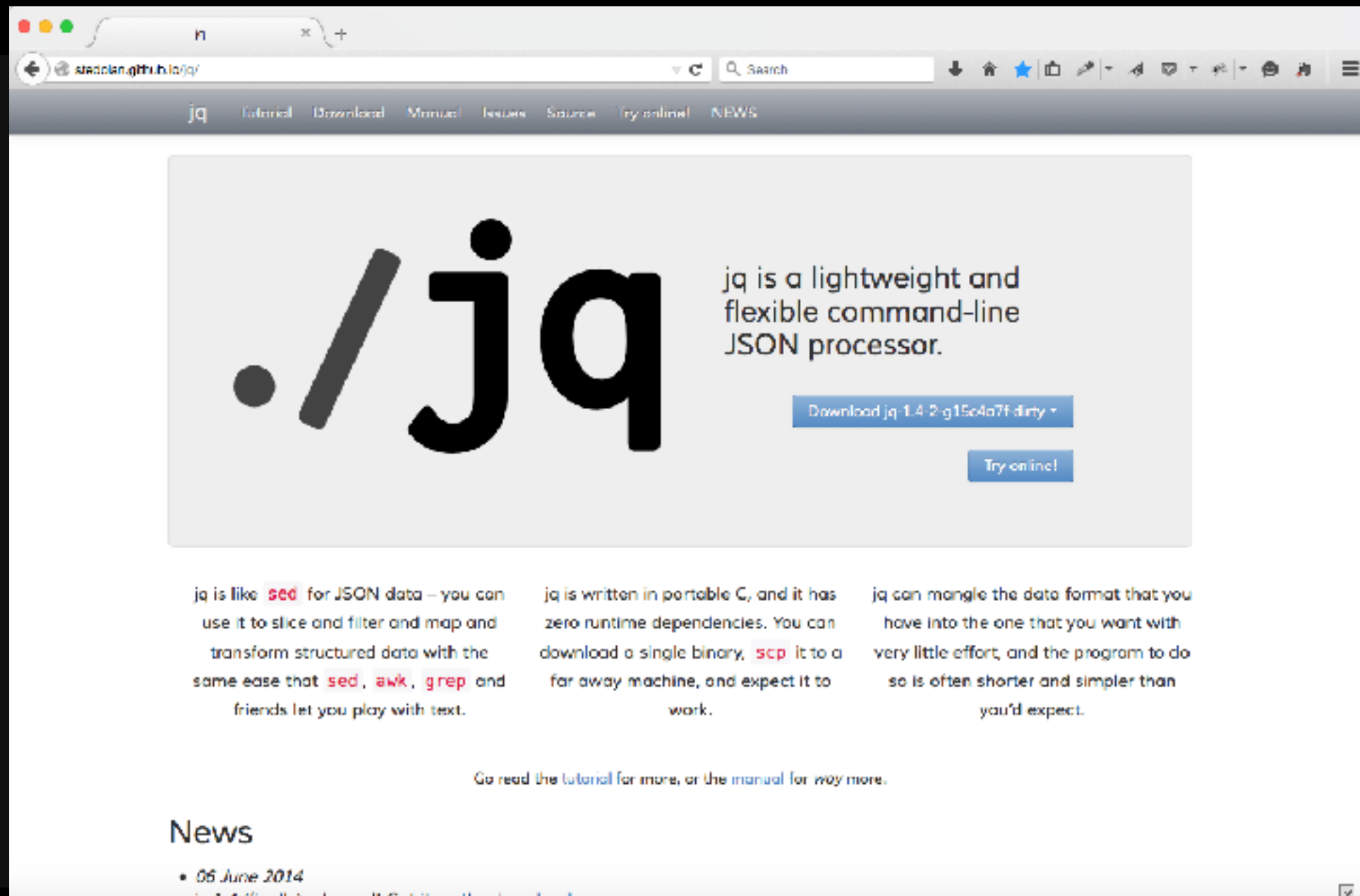
JSON Pointer Test

```
12 var expect = require('chai').expect;
13 var pointer = require('json-pointer');
14 var unirest = require('unirest');
15
16 describe('cities-json-pointer', function() {
17   var req;
18
19   beforeEach(function() {
20     req = unirest.get('http://localhost:5000/cities')
21       .header('Accept', 'application/json');
22   });
23
24   it('should return a 200 response', function(done) {
25     req.end(function(res) {
26       expect(res.statusCode).to.eql(200);
27       expect(res.headers['content-type']).to.eql(
28         'application/json; charset=utf-8');
29       done();
30     });
31   });
32
33   it('should return the 1st city', function(done) {
34     req.end(function(res) {
35       var cities = res.body;
36       var firstCity = null;
37
38       //console.log('\n\n1st Object: ');
39       firstCity = pointer.get(cities, '/0');
40       //console.log(firstCity);
41       expect(firstCity.name).to.eql('Rancho Palos Verdes');
42       expect(firstCity.weather[0].main).to.eql('Clear');
43       done();
44     });
45   });
46
47   it('should return the name of the 2nd city', function(done) {
```

JSON Pointer Scorecard

Mindshare	Y
Dev Community	Y
Platforms	JS, Node.js, Java, RoR
Intuitive	Y
Standard	RFC 6901 - Woot!

jq



The screenshot shows the homepage of the jq project. The browser's address bar displays 'stuartmorgan.github.io/jq/'. The navigation bar includes links for 'jq', 'Tutorial', 'Download', 'Manual', 'Issues', 'Source', 'Try online!', and 'NEWS'. The main content area features a large logo consisting of a dot, a slash, and the letters 'jq'. To the right of the logo, a text block states: 'jq is a lightweight and flexible command-line JSON processor.' Below this text are two buttons: 'Download jq 1.4 2-g15c4a7f-dirty' and 'Try online!'. Further down, three columns of text describe jq's capabilities: its similarity to sed for JSON, its portability and ease of distribution, and its ability to transform data formats. At the bottom, a 'News' section lists a release from June 2014.

jq

Tutorial Download Manual Issues Source Try online! NEWS

./jq

jq is a lightweight and flexible command-line JSON processor.

Download jq 1.4 2-g15c4a7f-dirty

Try online!

jq is like **sed** for JSON data – you can use it to slice and filter and map and transform structured data with the same ease that **sed**, **awk**, **grep** and friends let you play with text.

jq is written in portable C, and it has zero runtime dependencies. You can download a single binary, **scp** it to a far away machine, and expect it to work.

jq can mangle the data format that you have into the one that you want with very little effort, and the program to do so is often shorter and simpler than you'd expect.

Go read the [tutorial](#) for more, or the [manual](#) for way more.

News

- 06 June 2014
jq 1.4 (finally!) released! Get it on the [download page](#).

jq play

jqplay A playground for jq 1.4

Filter

JSON

Result

☐ Compact Output ☐ Null Input ☐ Raw Input ☐ Raw Output ☐ Slurp

Cheatsheet

Click on the icons (📄) in the table below to see examples.

.	unchanged input	📄	.	feed input into multiple filters	📄
.foo, .foo.bar, .foo?	value at key	📄		pipe output of one filter to the next filter	📄

jq Queries

2	<code>jq query</code>	Description
3	-----	
4	<code>.cities[0]</code>	Get the first city. <code>jq</code> Array filtering starts at 0.
5	<code>.cities[-1]</code>	Get the last city. An index of -1 indicates the last element of an Array.
6	<code>.cities[0:3]</code>	Get the first three cities, where 0 is the start index (inclusive), and 3 is the end index (exclusive).
7		
8		
9	<code>.cities[:3]</code>	Get the first three cities. This is shorthand, and it omits the start index.
10		
11	<code>[.cities[:3] .[] {name, weather}]</code>	
12		Get the name and weather for the first 3 cities and put it in a new array.
13		
14	<code>.cities[] select (.main.temp >= 80 and (.main.temp_min >= 79 and .main.temp_max <= 92))</code>	
15		Get all cities whose current temperature is >= 80 degrees Fahrenheit and whose
16		min and max temperature ranges between 79 and 92 degrees Fahrenheit
17		(inclusive).

jq Test

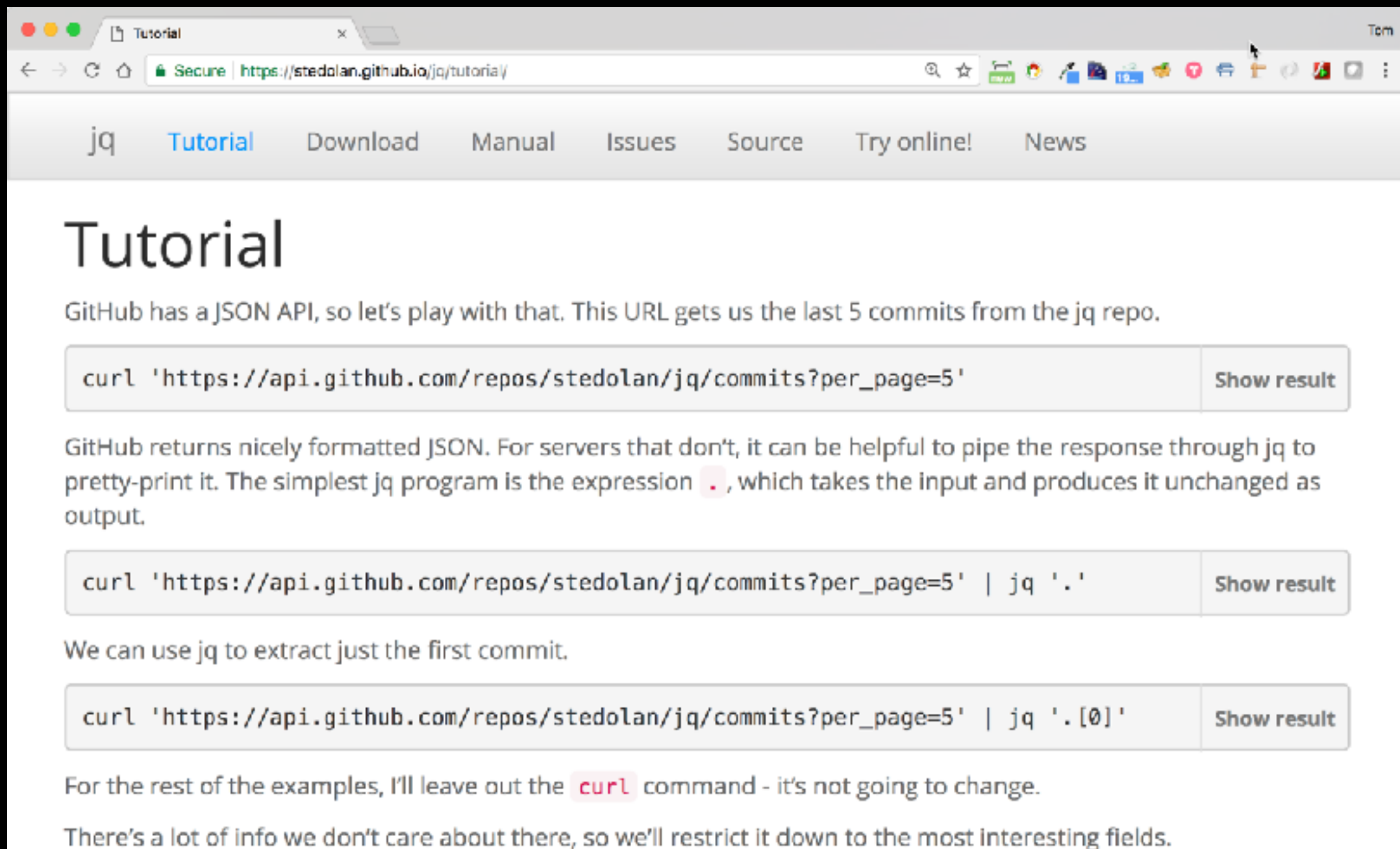
```
12  var expect = require('chai').expect;
13  var jq = require('node-jq');
14  var unirest = require('unirest');
15  var _ = require('underscore');
16
17
18  describe('cities-jq', function() {
19    var req;
20
21    beforeEach(function() {
22      req = unirest.get('http://localhost:5000/cities')
23        .header('Accept', 'application/json');
24    });
25
26    it('should return a 200 response', function(done) {
27      req.end(function(res) {
28        expect(res.statusCode).to.eql(200);
29        expect(res.headers['content-type']).to.eql(
30          'application/json; charset=utf-8');
31        done();
32      });
33    });
34
35    it('should return all cities', function(done) {
36      req.end(function(res) {
37        var cities = res.body;
38
39        //console.log(cities);
40        expect(cities.length).to.eql(110);
41        done();
42      });
43    });
44  });
```

jq and cURL

```
curl -X GET 'http://localhost:5000/cities'
```

```
curl -X GET 'http://localhost:5000/cities' | jq .[0]
```

jq Tutorial



The screenshot shows a web browser window with the title 'Tutorial' and the URL 'https://stedolan.github.io/jq/tutorial/'. The browser's address bar shows 'Secure' and the URL. The page has a navigation bar with links: 'jq', 'Tutorial' (highlighted), 'Download', 'Manual', 'Issues', 'Source', 'Try online!', and 'News'. The main content area has a large heading 'Tutorial' followed by a paragraph: 'GitHub has a JSON API, so let's play with that. This URL gets us the last 5 commits from the jq repo.' Below this is a code block containing a curl command and a 'Show result' button. The next paragraph explains that GitHub returns nicely formatted JSON and that the simplest jq program is the expression '.', which takes the input and produces it unchanged as output. This is followed by another code block with a curl command and a 'Show result' button. The next paragraph states: 'We can use jq to extract just the first commit.' This is followed by a third code block with a curl command and a 'Show result' button. The final paragraph says: 'For the rest of the examples, I'll leave out the curl command - it's not going to change.' The last line of the page says: 'There's a lot of info we don't care about there, so we'll restrict it down to the most interesting fields.'

jq **Tutorial** Download Manual Issues Source Try online! News

Tutorial

GitHub has a JSON API, so let's play with that. This URL gets us the last 5 commits from the jq repo.

```
curl 'https://api.github.com/repos/stedolan/jq/commits?per_page=5'
```

Show result

GitHub returns nicely formatted JSON. For servers that don't, it can be helpful to pipe the response through jq to pretty-print it. The simplest jq program is the expression `.`, which takes the input and produces it unchanged as output.

```
curl 'https://api.github.com/repos/stedolan/jq/commits?per_page=5' | jq '.'
```

Show result

We can use jq to extract just the first commit.

```
curl 'https://api.github.com/repos/stedolan/jq/commits?per_page=5' | jq '.[0]'
```

Show result

For the rest of the examples, I'll leave out the `curl` command - it's not going to change.

There's a lot of info we don't care about there, so we'll restrict it down to the most interesting fields.

jq-tutorial

```
json-at-work => jq-tutorial  
Run jq-tutorial with one of the following:  
  * pick  
  * objects  
  * mapping  
  * filtering  
  * output  
  * reduce
```

jq Scorecard

Mindshare	Y
Dev Community	Y
Platforms	CLI - Linux/Mac/ Windows, JS, Node.js, Java, RoR
Intuitive	Y
Standard	N

My JSON Search Choices

Tool	Rank
jq	1
JSONPath	2
JSON Pointer	3

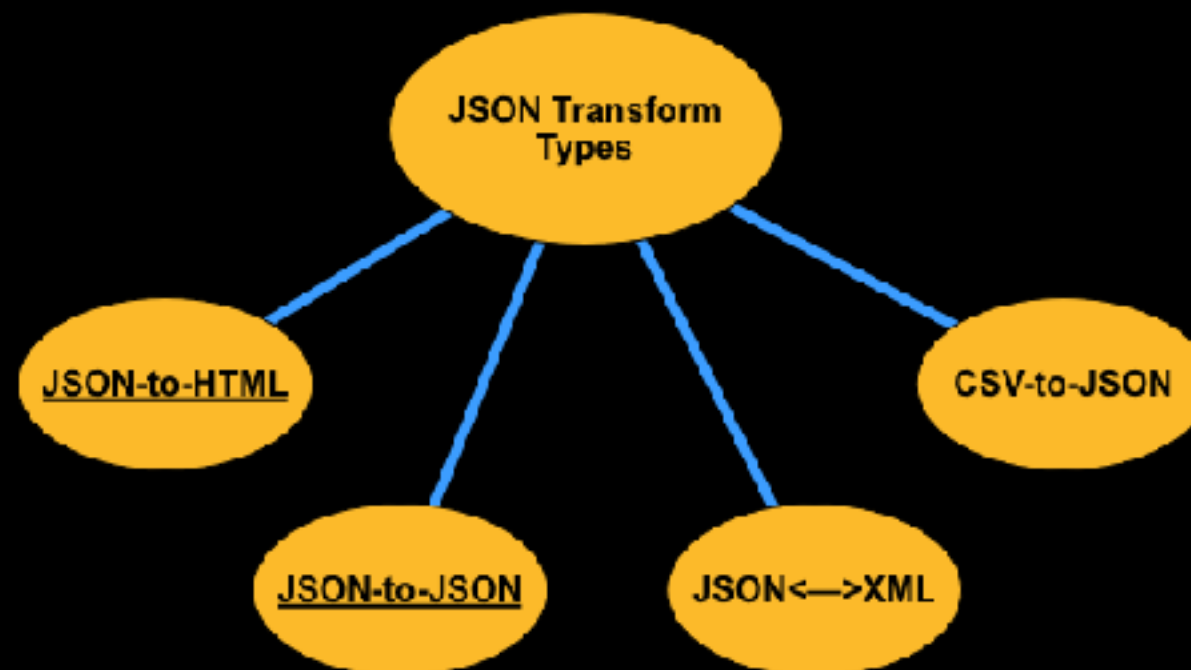
Where Are We?

JSON Search & Transform Overview	1
----------------------------------	---

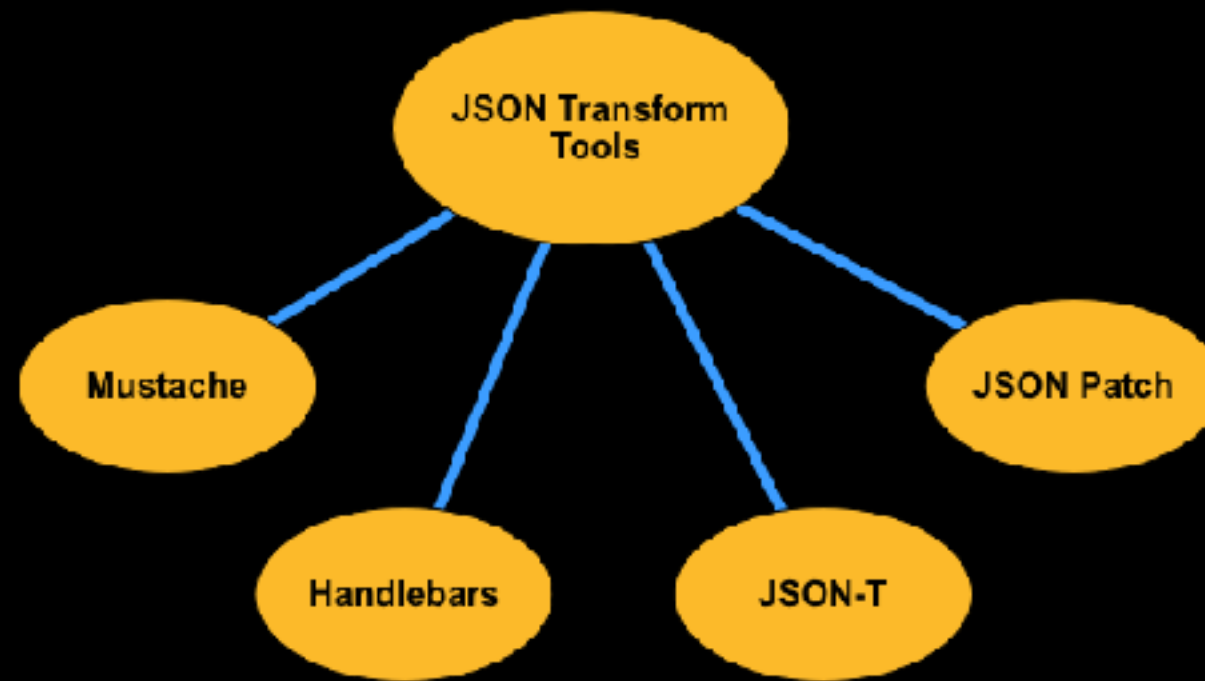
JSON Search	2
-------------	---

JSON Transform	3
----------------	---

JSON Transform Types



JSON Transform Tools



Architect - JS Template Editor

The screenshot shows the Architect JS Template Editor web application. The browser address bar displays `https://rownd.github.io/architect/`. The page title is "Architect" with the subtitle "Edit Javascript templates in various engines" and a status indicator "offline enabled".

At the top, there is a control bar with the following elements:

- Engine: `doT.js` (selected from a dropdown)
- Buttons: `Reset`, `version 1.0.1`, `size 1.5KB`, `github`, and `download`

The interface is divided into three main sections:

Template

```
1 <h1>{{=it.header}}</h1>
2
3 <select>
4   {{ for (var i=0; i < it.items.length; i++) { }}
5     <option value="{{=it.items[i].id}}"{{ if (it.items[i].active) { }} sel
6   {{ } }}
7 </select>
```

View

```
1 {
2   "header": "Colors",
3   "items": [
4     {"id": 1, "name": "Red"},
5     {"id": 2, "name": "Green", "active": true},
6     {"id": 3, "name": "Blue"}
7   ]
8 }
```

Result

```
1 <h1>Colors</h1><select> <option value="1">Red</option> <option value="2" selected>Green</option> <option value="3">Blue</option></select>
```

A diagonal banner in the top right corner reads "Fork me on GitHub".

Mustache

The screenshot shows the GitHub repository page for `mustache/mustache`. The browser address bar shows the URL `https://github.com/mustache/mustache`. The repository name is `mustache / mustache`. The page shows 47 watches, 2,411 stars, and 225 forks. The repository description is "Logic-less Ruby templates." with a link to `http://mustache.github.io/`. The repository statistics show 787 commits, 2 branches, 44 releases, 61 contributors, and the MIT license. The file list includes `locks v1.0.5`, `benchmarks`, `bin`, `examples`, `ext`, `lib`, `man`, `test`, `.gitignore`, and `.gitmodules`. The latest commit is `24a4cf6` on Mar 28.

mustache/mustache: Logic-less x

GitHub, Inc. [US] <https://github.com/mustache/mustache>

This repository Search Pull requests Issues Marketplace Explore

mustache / mustache Watch 47 Star 2,411 Fork 225

Code Issues 19 Pull requests 3 Projects 0 Wiki Insights

Logic-less Ruby templates. <http://mustache.github.io/>

787 commits 2 branches 44 releases 61 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

locks v1.0.5		2 Latest commit 24a4cf6 on Mar 28
benchmarks	Modified render_collection_benchmark to measure only rendering (not c...	2 years ago
bin	CLI option to raise exception on context misses	2 years ago
examples	self can use .	7 years ago
ext	Fix array of array iteration	3 years ago
lib	v1.0.5	6 months ago
man	fix data.yml template.mustache mustache example	10 months ago
test	Allow redefine initialize in Mustache descendants	6 months ago
.gitignore	Allow redefine initialize in Mustache descendants	6 months ago
.gitmodules	Updating ext/spec submodule pointer.	7 years ago

Mustache HTML Transform

Architect - Javascript Templating

Secure <https://rowno.github.io/architect/>

Architect Edit Javascript templates in various engines **offline enabled**

Engine: Mustache.js [Reset](#) [version 0.8.1](#) [size 2.0KB](#) [github](#) [download](#)

Template

```
30 <td>{{temp}}</td>
31 <td>{{temp_min}}</td>
32 <td>{{temp_max}}</td>
33 <td>{{humidity}}</td>
34 {{/main}}
35 <td>{{wind, speed}}</td>
36 {{#weather.0}}
37 <td>{{main}}</td>
38 <td>{{description}}</td>
39 {{/weather.0}}
40 </tr>
41 {{/cities}}
42 </table>
43 </body>
44
45 </html>
```

View

```
73 wind : {
74   "speed": 4.6,
75   "deg": 240
76 },
77 "clouds": {
78   "all": 32
79 },
80 "weather": [{
81   "id": 802,
82   "main": "Clouds",
83   "description": "scattered clouds",
84   "icon": "03d"
85 }]
86 }
87 }
88 }
```

Result

```
29 <td>78.8</td>
30 <td>93</td>
31 <td>58</td>
32 <td>4.1</td>
33 <td>Clear</td>
34 <td>Sky is Clear</td>
35 </tr>
36 <tr>
37 <td>5302528</td>
38 <td>San Pedro</td>
39 <td>84.02</td>
40 <td>78.8</td>
41 <td>91</td>
42 <td>58</td>
```

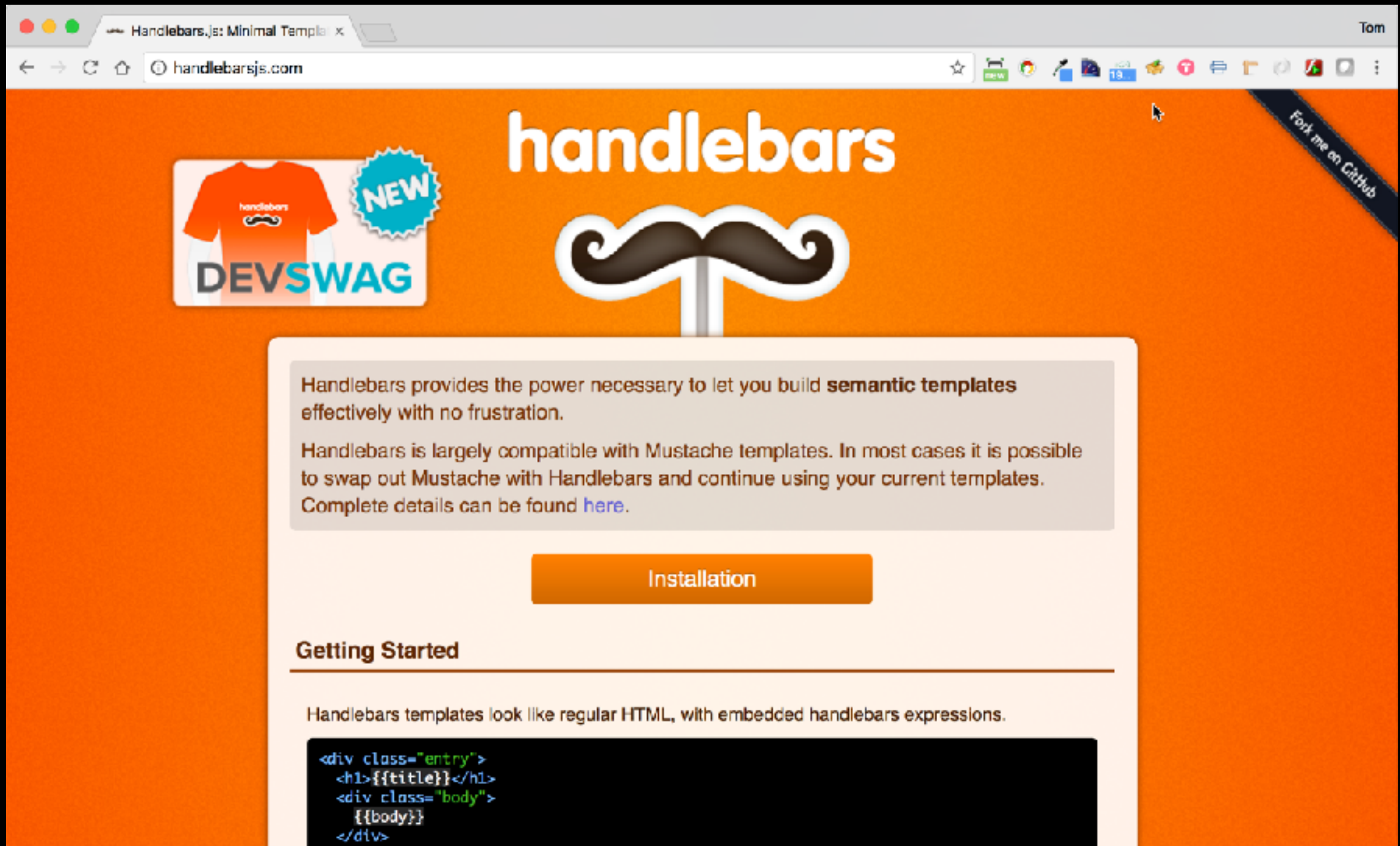
Mustache HTML Template

```
1 <!DOCTYPE html>
2 <html>
3
4   <head>
5     <meta charset="UTF-8" />
6     <title>OpenWeather - California Cities</title>
7     <link rel="stylesheet" href="weather.css">
8   </head>
9   <body>
10    <h1>OpenWeather - California Cities</h1>
11    <table class="weatherTable">
12      <thead>
13        <tr>
14          <th>ID</th>
15          <th>City</th>
16          <th>Current Temp</th>
17          <th>Low Temp</th>
18          <th>High Temp</th>
19          <th>Humidity</th>
20          <th>Wind Speed</th>
21          <th>Summary</th>
22          <th>Description</th>
23        </tr>
24      </thead>
25      {{#cities}}
26        <tr>
27          <td>{{id}}</td>
28          <td>{{name}}</td>
29          {{#main}}
30            <td>{{temp}}</td>
31            <td>{{temp_min}}</td>
32            <td>{{temp_max}}</td>
33            <td>{{humidity}}</td>
34          {{/main}}
35          <td>{{wind.speed}}</td>
36          {{#weather.0}}
37            <td>{{main}}</td>
38            <td>{{description}}</td>
39          {{/weather.0}}
```

Mustache HTML Test

```
12 var expect = require('chai').expect;
13 var jsonfile = require('jsonfile');
14 var fs = require('fs');
15 var mustache = require('mustache');
16
17 describe('cities-mustache', function() {
18   var jsonCitiesFileName = null;
19   var htmlTemplateFileName = null;
20
21   beforeEach(function() {
22     var baseDir = __dirname + '/../..';
23
24     jsonCitiesFileName = baseDir + '/data/cities-weather-short.json';
25     htmlTemplateFileName = baseDir +
26       '/templates/transform-html.mustache';
27   });
28
29   it('should transform cities JSON data to HTML', function(done) {
30     jsonfile.readFile(jsonCitiesFileName, function(readJsonFileError,
31       jsonObj) {
32       if (!readJsonFileError) {
33         fs.readFile(htmlTemplateFileName, 'utf8', function(
34           readTemplateFileError, templateFileData) {
35             if (!readTemplateFileError) {
36               var template = templateFileData.toString();
37               var html = mustache.render(template, jsonObj);
38
39               console.log('\n\n\nHTML Output:\n' + html);
40               done();
41             } else {
42               done(readTemplateFileError);
43             }
44           });
45       } else {
46         done(readJsonFileError);
47       }
48     });
49   });
50 });
```


Handlebars



A screenshot of a web browser displaying the Handlebars.js website. The browser's address bar shows 'handlebarsjs.com'. The website has an orange background. On the left, there is a logo featuring a red t-shirt with a mustache and the text 'DEVSWAG' and 'NEW'. In the center, the word 'handlebars' is written in white, with a large white mustache graphic below it. On the right, a diagonal banner says 'Fork me on GitHub'. Below the main header, there is a white box containing text about semantic templates and compatibility with Mustache. An orange button labeled 'Installation' is below this text. Further down, a section titled 'Getting Started' contains a paragraph about Handlebars templates and a code block showing HTML with embedded Handlebars expressions.

Handlebars provides the power necessary to let you build **semantic templates** effectively with no frustration.

Handlebars is largely compatible with Mustache templates. In most cases it is possible to swap out Mustache with Handlebars and continue using your current templates. Complete details can be found [here](#).

Installation

Getting Started

Handlebars templates look like regular HTML, with embedded handlebars expressions.

```
<div class="entry">
  <h1>{{title}}</h1>
  <div class="body">
    {{body}}
  </div>
  ...
</div>
```

Handlebars HTML Transform

Architect - Javascript Template Editor

Secure | <https://rowno.github.io/architect/>

Architect Edit Javascript templates in various engines **offline enabled**

Engine: Mustache.js Reset version 0.8.1 size 2.0KB [github](#) [download](#)

Template

```
30 <td>{{temp}}</td>
31 <td>{{temp_min}}</td>
32 <td>{{temp_max}}</td>
33 <td>{{humidity}}</td>
34 {{/main}}
35 <td>{{wind.speed}}</td>
36 {{#weather.0}}
37 <td>{{main}}</td>
38 <td>{{description}}</td>
39 {{/weather.0}}
40 </tr>
41 {{/cities}}
42 </table>
43 </body>
44
45 </html>
```

View

```
73 =
74   "speed": 4.6,
75   "deg": 240
76 },
77 "clouds": {
78   "all": 32
79 },
80 "weather": [{
81   "id": 802,
82   "main": "Clouds",
83   "description": "scattered clouds",
84   "icon": "03d"
85 }]
86 ]]
87 }
88
```

Result

```
28 <td>78.8</td>
29 <td>78.8</td>
30 <td>93</td>
31 <td>58</td>
32 <td>4.1</td>
33 <td>Clear</td>
34 <td>Sky is Clear</td>
35 </tr>
36 <tr>
37 <td>5392528</td>
38 <td>San Pedro</td>
39 <td>84.02</td>
40 <td>78.8</td>
41 <td>91</td>
42 <td>58</td>
```


Handlebars HTML Template

```
1  <!DOCTYPE html>
2  <html>
3
4    <head>
5      <meta charset="UTF-8" />
6      <title>OpenWeather - California Cities</title>
7      <link rel="stylesheet" href="weather.css">
8    </head>
9    <body>
10     <h1>OpenWeather - California Cities</h1>
11     <table class="weatherTable">
12       <thead>
13         <tr>
14           <th>ID</th>
15           <th>City</th>
16           <th>Current Temp</th>|
17           <th>Low Temp</th>
18           <th>High Temp</th>
19           <th>Humidity</th>
20           <th>Wind Speed</th>
21           <th>Summary</th>
22           <th>Description</th>
23         </tr>
24       </thead>
25       {{#each cities}}
```

Handlebars HTML Test

```
12 var expect = require('chai').expect;
13 var jsonfile = require('jsonfile');
14 var fs = require('fs');
15 var handlebars = require('handlebars');
16
17 describe('cities-handlebars', function() {
18   var jsonCitiesFileName = null;
19   var htmlTemplateFileName = null;
20
21   beforeEach(function() {
22     var baseDir = __dirname + '/../..';
23
24     jsonCitiesFileName = baseDir + '/data/cities-weather-short.json';
25     htmlTemplateFileName = baseDir +
26       '/templates/transform-html.hbs';
27   });
28
29   it('should transform cities JSON data to HTML', function(done) {
30     jsonfile.readFile(jsonCitiesFileName, function(readJsonFileError,
31       jsonObj) {
32       if (!readJsonFileError) {
33         fs.readFile(htmlTemplateFileName, 'utf8', function(
34           readTemplateFileError, templateFileData) {
35             if (!readTemplateFileError) {
36               var template = handlebars.compile(templateFileData);
37               var html = template(jsonObj);
38
39               console.log('\n\nHTML Output:\n' + html);
40               done();
41             } else {
42               done(readTemplateFileError);
43             }
44           });
45       } else {
46         done(readJsonFileError);
47       }
48     });
49   });
50 });
```

Mustache JSON Transform

The screenshot shows the Architect - Javascript Template Editor interface. The browser address bar displays <https://rowno.github.io/architect/>. The interface is divided into three main sections: Template, View, and Result.

Template

```
1 {{#main}}
2   "currentTemp": {{temp}},
3   "lowTemp": {{temp_min}},
4   "hiTemp": {{temp_max}},
5   "humidity": {{humidity}},
6 {{/main}}
7 "windSpeed": {{wind.speed}},
8 {{#weather.0}}
9   "summary": "{{main}}"
10  "description": "{{description}}"
11 {{/weather.0}}
12 }
13 },
14 {{/cities}}
15 ]
16 }
```

View

```
1 {
2   "cities": [{
3     "id": 5386035,
4     "name": "Rancho Palos Verdes",
5     "coord": {
6       "lon": -118.387016,
7       "lat": 33.744461
8     },
9     "main": {
10      "temp": 84.34,
11      "pressure": 1012,
12      "humidity": 58,
13      "temp_min": 78.8,
14      "temp_max": 93
15    },
16    "dt": 1447171078
17  }
18 ]
19 }
```

Result

```
1 {
2   "id": "3988392",
3   "name": "Rosarito",
4   "weather": {
5     "currentTemp": 82.47,
6     "lowTemp": 78.8,
7     "hiTemp": 86,
8     "humidity": 61,
9     "windSpeed": 4.6,
10    "summary": "Clouds"
11    "description": "scattered clouds"
12  }
13 },
14 ]
15 }
```

At the bottom of the editor, there is a footer with the GitHub logo, a settings icon, a refresh icon, and a copyright notice: Copyright © 2012 Roland Warmerdam. Released under the MIT license.

Mustache JSON Template

```
1  {
2    "cities": [
3      {{#cities}}
4        {
5          "id": "{{id}}",
6          "name": "{{name}}",
7          "weather": {
8            {{#main}}
9              "currentTemp": {{temp}},
10             "lowTemp": {{temp_min}},
11             "hiTemp": {{temp_max}},
12             "humidity": {{humidity}},
13             {{/main}}
14             "windSpeed": {{wind.speed}},
15             {{#weather.0}}
16             "summary": "{{main}}"
17             "description": "{{description}}"
18             {{/weather.0}}
19           }
20         },
21       {{/cities}}
22     ]
23   }
```

Handlebars JSON Transform

Architect - Javascript Template X

Secure | <https://rowno.github.io/architect/>

Architect Edit Javascript templates in various engines **offline enabled**

Engine: [Reset](#) [version 1.3.0](#) [size 13.9KB](#) [github](#) [download](#)

Template

```
1- {{#main}}
2-   "currentTemp": {{temp}},
3-   "lowTemp": {{temp_min}},
4-   "hiTemp": {{temp_max}},
5-   "humidity": {{humidity}},
6-   {{/main}}
7-   "windSpeed": {{wind.speed}},
8-   {{#each weather}}
9-   "summary": "{{{main}}}",
10-  "description": "{{{description}}}"
11-  {{/each}}
12-  }
13-  {{#unless @last}},{{/unless}}
14-  {{/each}}
15- ]
16- }
```

View

```
1- {
2-   "cities": [{
3-     "id": 5386035,
4-     "name": "Rancho Palos Verdes",
5-     "coord": {
6-       "lon": -118.387016,
7-       "lat": 33.744461
8-     },
9-     "main": {
10-      "temp": 84.34,
11-      "pressure": 1012,
12-      "humidity": 58,
13-      "temp_min": 78.8,
14-      "temp_max": 93
15-    },
16-     "dt": 1442171078
17-   },
18-   ...
19- ]
20- }
```

Result

```
1- {
2-   "currentTemp": 82.47,
3-   "lowTemp": 78.8,
4-   "hiTemp": 86,
5-   "humidity": 61,
6-   "windSpeed": 4.5,
7-   "summary": "Clouds",
8-   "description": "scattered clouds"
9- }
10- }
```

Handlebars JSON Template

```
1  {
2    "cities": [
3      {{#each cities}}
4        {
5          "id": "{{id}}",
6          "name": "{{name}}",
7          "weather": {
8            {{#main}}
9              "currentTemp": {{temp}},
10             "lowTemp": {{temp_min}},
11             "hiTemp": {{temp_max}},
12             "humidity": {{humidity}},
13            {{/main}}
14            "windSpeed": {{wind.speed}},
15            {{#each weather}}
16              "summary": "{{main}}",
17              "description": "{{description}}"
18            {{/each}}
19          }
20        }{{#unless @last}},{{/unless}}
21      {{/each}}
22    ]
23  }
```

Agenda

JSON Search & Transform Overview	1
---	----------

JSON Search	2
--------------------	----------

JSON Transform	3
-----------------------	----------

What's The Point?

JSON Search and Transform

Simplify interaction with RESTful APIs

Your Takeaway

jq + Handlebars Rock!

Questions?

Tom Marrs

thomasamarrs@comcast.net



JSON Resources

JSON Generator - <http://www.json-generator.com/>

json.org - <http://www.json.org>

JSON Editor Online - <http://jsoneditoronline.org/>

JSONPath Resources

<http://goessner.net/articles/JsonPath/>

<https://github.com/jayway/JsonPath>

[https://rubygems.org/gems/jsonpath/versions/
0.5.6](https://rubygems.org/gems/jsonpath/versions/0.5.6)

<https://www.npmjs.com/package/jsonpath>

jq Resources

<http://stedolan.github.io/jq/>

<http://stedolan.github.io/jq/tutorial/>

<https://github.com/stedolan/jq>

<https://robots.thoughtbot.com/jq-is-sed-for-json>

<https://zerokspot.com/weblog/2013/07/18/processing-json-with-jq/>

<https://jqplay.org/>

JSON Groups

Google - <http://groups.google.com/group/json-schema>

Yahoo! - <http://tech.groups.yahoo.com/group/json/>