

JSON at Work: Search and Transform

Tom Marrs



About Me ...



What's The Point?

JSON Search and Transform ...

Simplify interaction with RESTful APIs

Our Agenda

Overview

JSON Search

JSON Transform

We're Not Covering :-

REST

Deep JS

Other Languages

Examples and Slides...

<https://github.com/tmarrs/presentations/tree/master/JSON-at-Work-Search-and-Transform>

Where Are We?

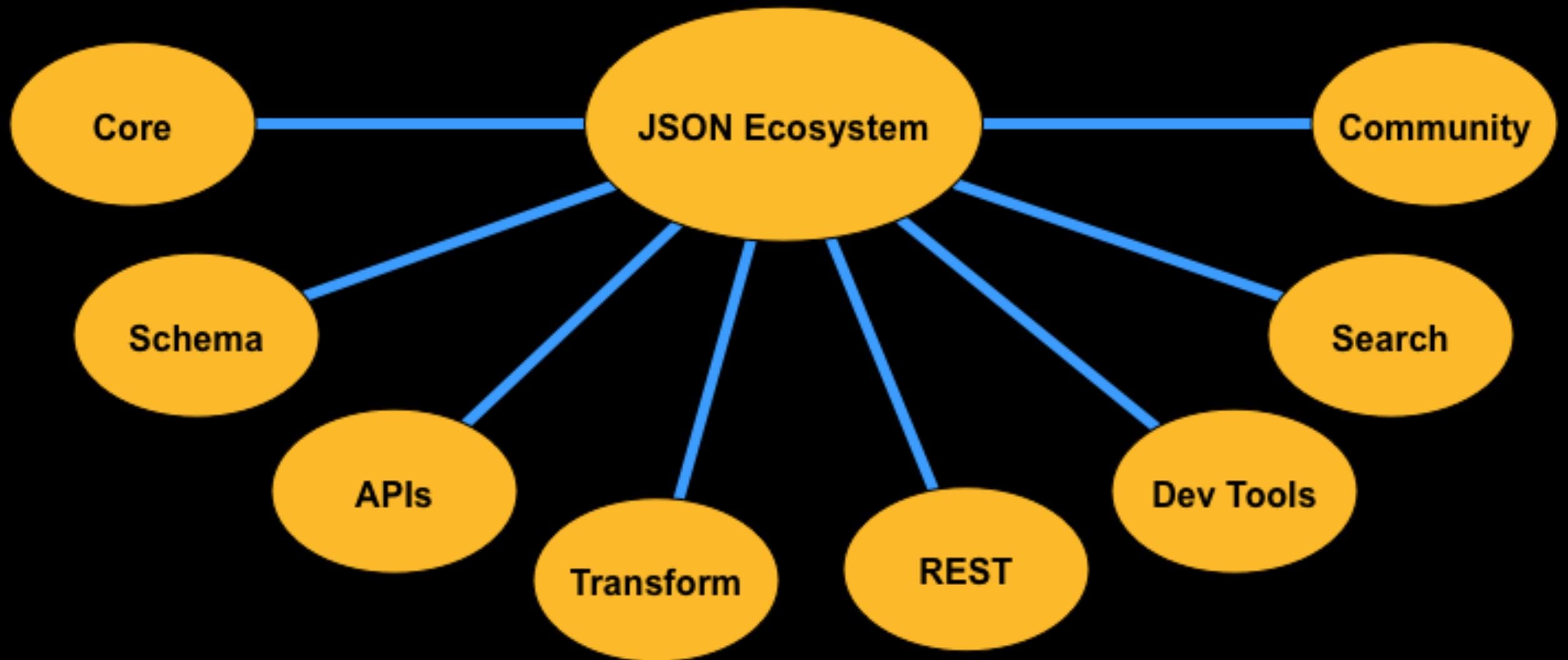
Overview

JSON Search

JSON Transform

JSON X

JSON Ecosystem

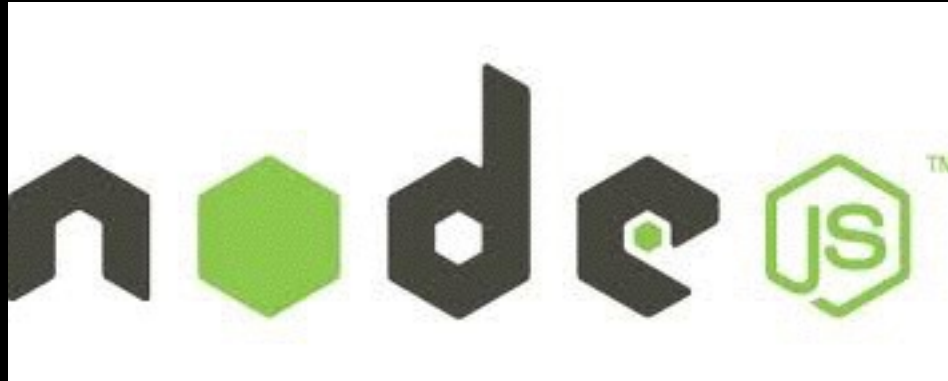


A Long Time Ago ...

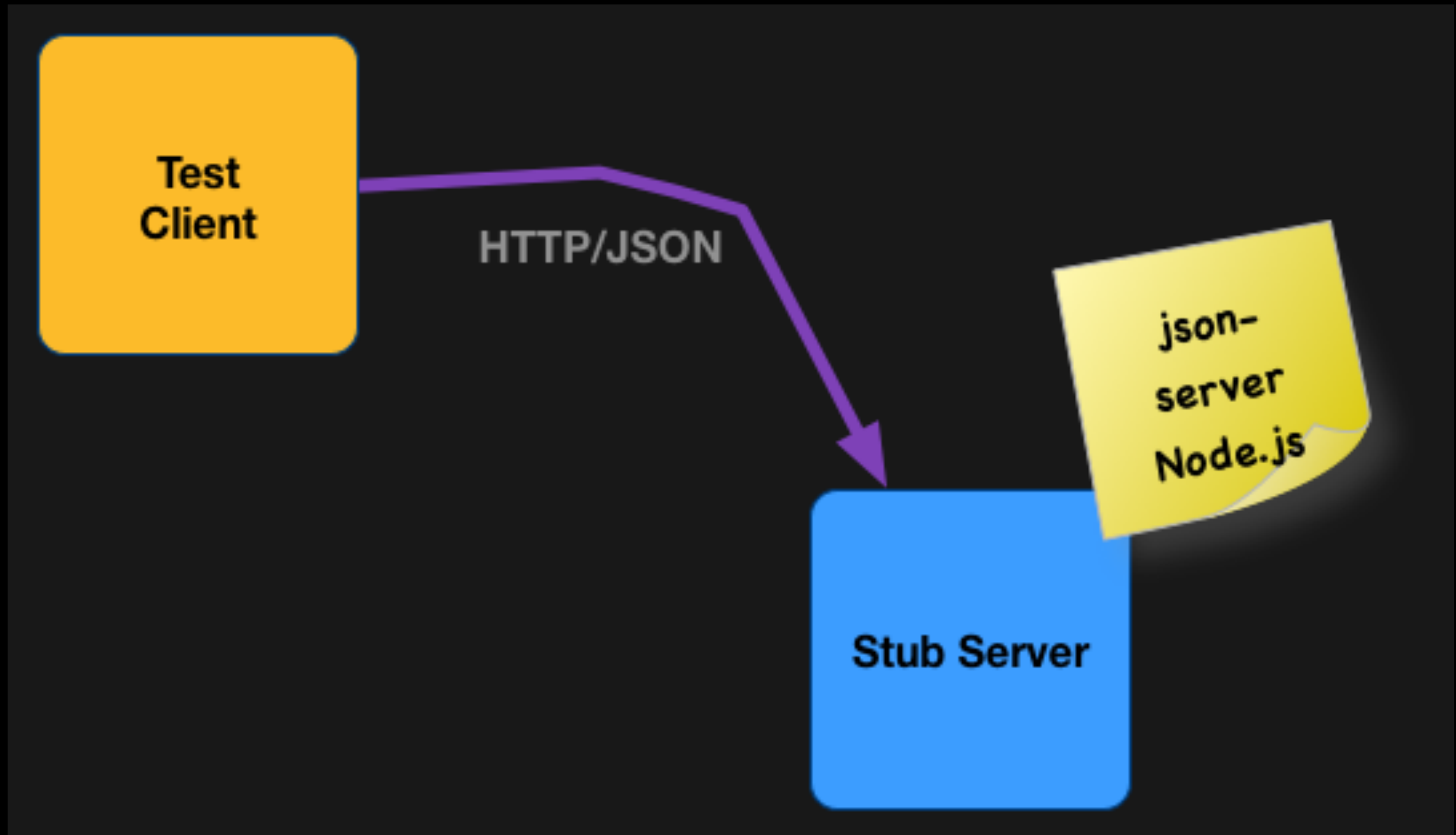
Algorithms + Data Structures = Programs

– Nicklaus Wirth

Our Client Stack



Our Architecture



Firebase Open Data Set

The screenshot shows the Firebase Open Data Sets documentation page. The browser address bar displays `https://www.firebase.com/docs/open-data/`. The page features a top navigation bar with the Firebase logo, links for OVERVIEW, GETTING STARTED, PRICING, and DOCS (which is highlighted), and buttons for LOGIN and SIGN UP. A left sidebar contains a search bar and a list of navigation items: Docs Home, PLATFORMS (Web, iOS, Android, REST), Security & Rules, Hosting, Partnerships, and Open Data Sets (which is selected). Below the Open Data Sets link in the sidebar, a list of data sets is shown: Airport Delays, Cryptocurrencies, Earthquakes, and Parking. The main content area has a blue header with the text 'FIREBASE Open Data Sets'. Below this, a paragraph explains that these data sets are populated with live-updated data from various sources. At the bottom, a table lists the available data sets and their descriptions.

Data Set	Description
Airport Delays	Get the latest airport delay and status updates in realtime.
Cryptocurrencies	Get the latest USD/BTC and USD/LTC exchange rates in realtime.
Earthquakes	Information on earthquakes anywhere on Earth in realtime.
Parking	Realtime data on the latest street parking price and garage availability for SF.

json-server on GH

The screenshot shows the GitHub repository page for `typicode/json-server`. The browser address bar shows the URL `https://github.com/typicode/json-server`. The repository name `typicode / json-server` is displayed at the top, along with statistics: 29 watchers, 878 stars, and 64 forks. Below this, a description reads: "Get a full fake REST API with zero coding in less than 30 seconds (seriously)".

Repository statistics are shown: 193 commits, 1 branch, 43 releases, and 10 contributors. The current branch is `master`. The version `0.7.0` is highlighted.

A commit by `typicode` from 2 days ago is shown, with the latest commit hash `d9b1eebbff`. The commit message is "Allow middlewares to be set before json-server middlewares". The files changed in this commit are:

File	Change	Time
<code>bin</code>	Allow middlewares to be set before json-server middlewares	2 days ago
<code>src</code>	Allow middlewares to be set before json-server middlewares	2 days ago
<code>test</code>	Allow middlewares to be set before json-server middlewares	2 days ago
<code>.gitignore</code>	Fix CLI	2 months ago
<code>.travis.yml</code>	Update travis.yml to Node 0.12	2 months ago
<code>LICENSE</code>	update copyright year	a year ago
<code>README.md</code>	Allow middlewares to be set before json-server middlewares	2 days ago
<code>package.json</code>	0.7.0	2 days ago

Below the commit list, the `README.md` file is selected. The README content shows the title **JSON Server** and two status badges: `build passing` and `npm package 0.7.0`.

On the right side, the `Code` tab is active, showing options to clone the repository. The HTTPS clone URL is `https://github.com/1`. Below this, there are buttons for `Clone in Desktop` and `Download ZIP`.

Install and Run ...

```
$ npm install -g json-server
```

```
$ json-server -p ./airports.json
```

Our Stub Server

```
{
  {
    id: 1,
    IATA: "ATL",
    ICAO: "KATL",
    city: "Atlanta",
    delay: true,
    name: "The William B Hartsfield International",
    state: "Georgia",
    status: {
      avgDelay: "",
      closureBegin: "",
      closureEnd: "",
      endTime: "",
      maxDelay: "30 minutes",
      minDelay: "16 minutes",
      reason: "WX:Wind",
      trend: "Increasing",
      type: "Arrival"
    },
    weather: {
      temp: "68.0 F (20.0 C)",
      visibility: 9,
      weather: "Light Rain",
      wind: "South at 12.7mph"
    }
  },
  {
    id: 2,
    IATA: "BNA",
    ICAO: "KBNA",
    city: "Nashville",
    delay: false,
    name: "Nashville International",
    state: "Tennessee",
    status: {
      avgDelay: "",
      closureBegin: "",
      closureEnd: "",
      endTime: "",
      maxDelay: "",
      minDelay: "",
      reason: "No known delays for this airport.",
      trend: "",
      type: ""
    },
    weather: {
      temp: "69.0 F (20.6 C)",
      visibility: 4,
      weather: "Thunderstorm Light Rain Fog/Mist",

```


JSON Search & Transform Rubrik



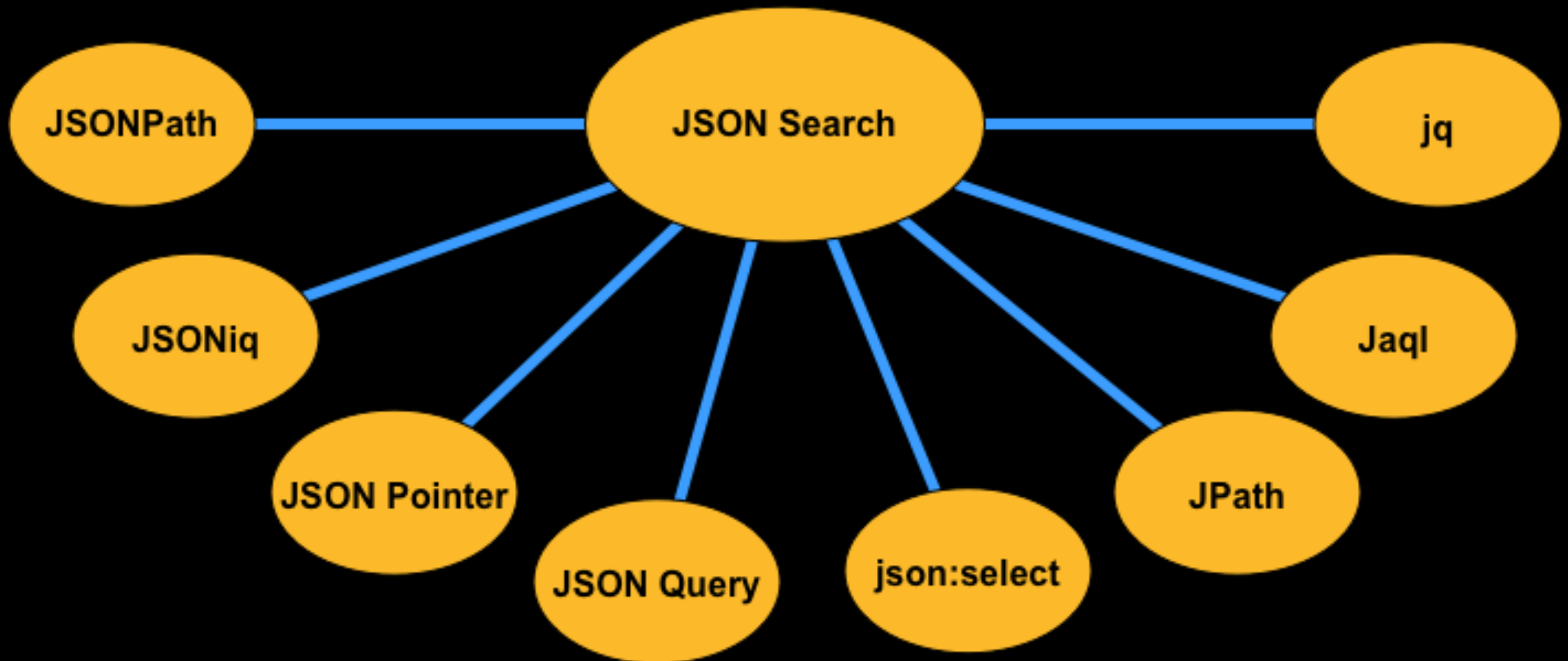
Where Are We?

Overview

JSON Search

JSON Transform

JSON Search



JSONPath

The screenshot shows a web browser window with the address bar displaying 'goessner.net/articles/JsonPath/'. The page title is '<stefan.goessner/> Mechanik, das Web und der ganze Rest'. The navigation bar includes links for 'Home', 'Lehre', 'Download', and 'Info'. The main content area is titled '# JSONPath - XPath for JSON' with a date of '2007-02-21' and a version 'e1'. The text discusses the advantages of XML and XPath, then introduces JSONPath as a tool for extracting data from JSON structures. It provides examples of XPath expressions and their JSONPath equivalents. A sidebar on the right contains a search bar, a table of contents under '» Inhalt ..', and a section for '» comments ..'.

<stefan.goessner/>
Mechanik, das Web und der ganze Rest

[Home](#) | [Lehre](#) | [Download](#) | [Info](#)

JSONPath - XPath for JSON

| 2007-02-21 | e1

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

» 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 "JSONPATH" logo and navigation links for "About", "Learn", "Changelog", and "Contact". The main content area has a green background and contains the following elements:

- JSON Data/URL**: A large white text area for inputting JSON data or a URL.
- JSONPath Expression**: A white text area for inputting the JSONPath expression.
- Process**: A green button located below the JSONPath Expression input.
- JSON Template**: A dropdown menu currently set to "2 Space Tab".
- Implementation**: A dropdown menu currently set to "JSONPath 0.8.3".
- Instructions**: On the left, a curved arrow points to the input fields with the text: "Paste in JSON or a URL, enter the JSONPath and away you go."
- Output**: On the right, a curved arrow points to the output area, which is currently empty.

JSONPath Demo

JSONPath Testing

```
1  var expect = require('chai').expect;
2  var request = require('request');
3  var jp = require('jsonpath');
4
5  describe('json-pointer', function() {
6    describe('api', function() {
7      it('should return 200', function(done) {
8        var options = {
9          url: 'http://localhost:5000/airports',
10         headers: {
11           'Content-Type': 'application/json'
12         }
13       };
14       request.get(options, function(err, res, body) {
15         expect(res.statusCode).to.equal(200);
16         console.log('\n\n\nJSONPath Test');
17         //console.log(res.body);
18         //var obj = JSON.parse(res.body);
19         var obj = JSON.parse(res.body);
20         console.log('\n\n1st & 3rd Object weather: ');
21         console.log(jp.query(obj, '$[0,2].weather'));
22
23         done();
24       });
25     });
26   });
27
28 });
```


JSONPath Scorecard

Mind Share	Y?
Dev Community	Y?
Platforms	JavaScript, Node.js, Java, Ruby
Intuitive	Y
Standard	N

JSONiq



The screenshot shows the homepage of the JSONiq website. At the top, there's a browser window with the address bar showing 'www.jsoniq.org'. Below the browser window is a navigation bar with links: 'The JSONiq Language', 'JSONiq Book', 'JSONiq Extension to XQuery', 'Implementations', 'Live Sandbox', and a 'Mailing-List' button. The main header features a large JSON object: `{ "codename" : ["JSONiq"] }`. Below this, the title 'The JSON Query Language' is centered. The page is divided into three columns, each with an icon and a heading. The first column has a 'Hello my name is JSONiq' graphic and the heading 'Decades of Lessons Learnt'. The second column has a gear icon and the heading 'Complex Processing'. The third column has a cloud and database icon and the heading 'The SQL of NoSQL'. Each column contains a paragraph of text describing JSONiq's capabilities and history.

JSONiq - The JSON Query Lang... ✕ +

www.jsoniq.org

The JSONiq Language JSONiq Book JSONiq Extension to XQuery Implementations Live Sandbox Mailing-List

```
{ "codename" : [ "JSONiq" ] }
```

The JSON Query Language



Decades of Lessons Learnt

JSONiq is a query and processing language specifically designed for the popular JSON data model. The main ideas behind JSONiq are based on lessons learnt in more than 30 years of relational query systems and more than 15 years of experience with designing and implementing query languages for semi-structured data.



Complex Processing

A JSONiq program is an expression; the result of the program is the result of the evaluation of the expression. Expressions have fundamental role in the language: every language construct is an expression, and expressions are fully composable. Project, Filter, Join, Group... Like SQL, JSONiq can do all that.



The SQL of NoSQL

JSONiq is an expressive and highly optimizable language to query and update NoSQL stores. It enables developers to leverage the same productive high-level language across a variety of NoSQL products.

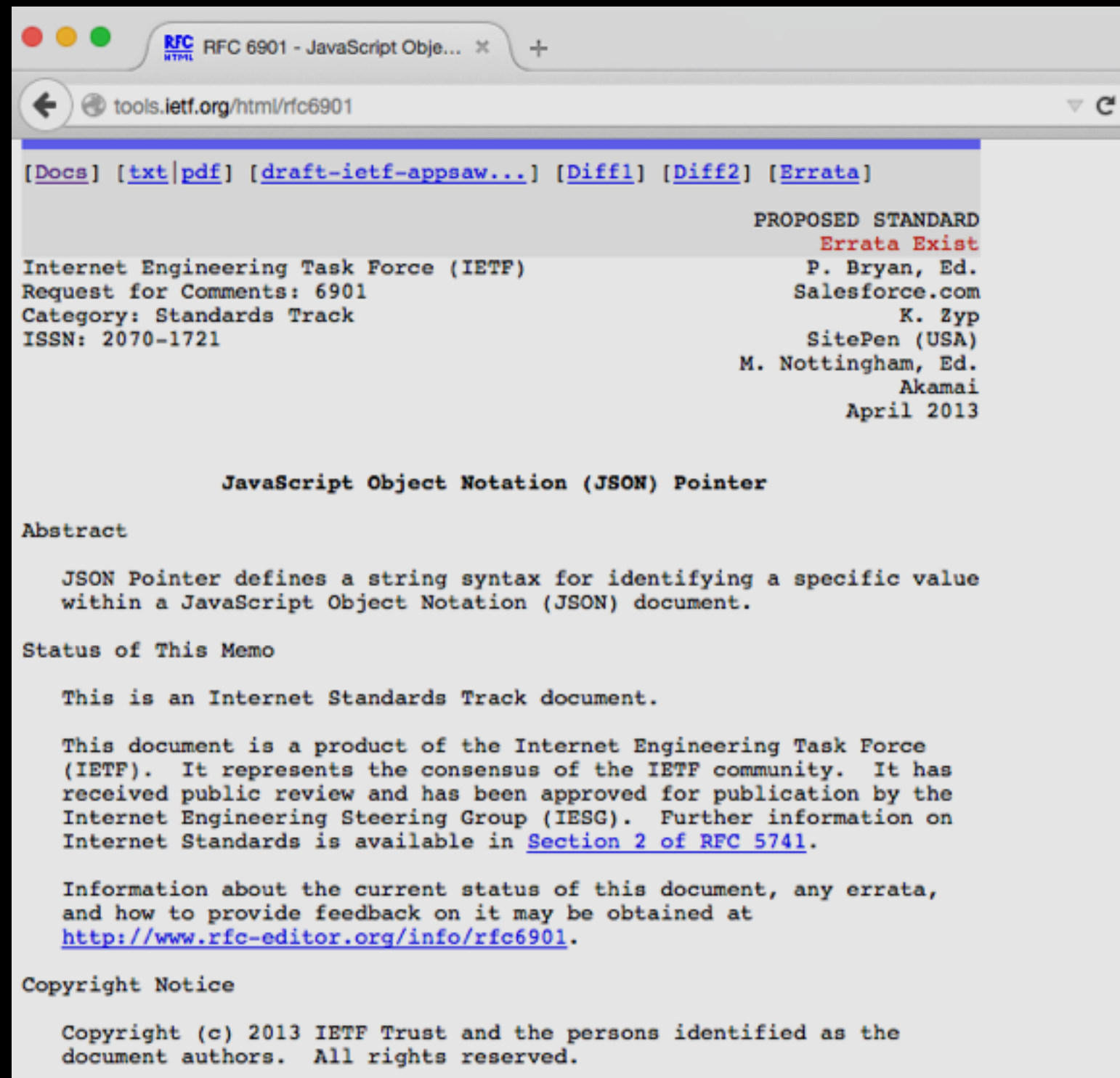
JSONiq Example

```
1
2 42 instance of integer # true
3 42 instance of decimal # true
4 42.6 instance of decimal # true
5 42.6e10 instance of double # true
6 "fred" instance of string # true
7 true instance of boolean # true
8 null instance of null # true
9
10
11 "Douglas" || " " || "Crockford", # Douglas Crockford
12 "Douglas" || () || "Crockford"    # DouglasCrockford
13
14 [ "question", "answer" ][][1] # "question"
15
16 {
17   questions: [
18     "What JSON Search too should I use?",
19     { "faq" : "We're still figuring out out." }
20   ]
21 }.questions[][2].faq           # "We're still figuring out out."
22
```

JSONiq Scorecard

Mind Share	N?
Dev Community	Y?
Platforms	Zorba.io
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 title is "RFC 6901 - JavaScript Object Notation (JSON) Pointer". The page content includes navigation links at the top: [\[Docs\]](#), [\[txt\]](#), [\[pdf\]](#), [\[draft-ietf-appsaw...\]](#), [\[Diff1\]](#), [\[Diff2\]](#), and [\[Errata\]](#). Below these links, the document is identified as a "PROPOSED STANDARD" with "Errata Exist". The Internet Engineering Task Force (IETF) information is listed: "Request for Comments: 6901", "Category: Standards Track", and "ISSN: 2070-1721". The authors and editors are listed as "P. Bryan, Ed.", "Salesforce.com", "K. Zyp", "SitePen (USA)", "M. Nottingham, Ed.", and "Akamai", with the date "April 2013". The title "JavaScript Object Notation (JSON) Pointer" is centered. The "Abstract" section 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 states: "This is an Internet Standards Track document." and "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](#)." The "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>." The "Copyright Notice" section states: "Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved."

[\[Docs\]](#) [\[txt\]](#) [\[pdf\]](#) [\[draft-ietf-appsaw...\]](#) [\[Diff1\]](#) [\[Diff2\]](#) [\[Errata\]](#)

PROPOSED STANDARD
Errata Exist

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

P. Bryan, Ed.
Salesforce.com
K. 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 Testing

```
1
2 var expect = require('chai').expect;
3 var request = require('request');
4 var pointer = require('json-pointer');
5
6 describe('json-pointer', function() {
7   describe('api', function() {
8     it('should return 200', function(done) {
9       var options = {
10         url: 'http://localhost:5000/airports',
11         headers: {
12           'Content-Type': 'application/json'
13         }
14       };
15       request.get(options, function(err, res, body) {
16         expect(res.statusCode).to.equal(200);
17         //console.log(res.body);
18         var obj = JSON.parse(res.body);
19         console.log('\n\n\nJSON Pointer Test');
20         console.log('\n\n1st Object: ');
21         console.log(pointer.get(obj, '/0'));
22         console.log('\nIATA on 2nd Object: ');
23         console.log(pointer.get(obj, '/1/IATA'));
24         done();
25       });
26     });
27   });
28 });
```

JSON Pointer Scorecard

Mind Share	Y?
Dev Community	Y
Platforms	JavaScript, Node.js, Java, Ruby, etc.
Intuitive	Y
Standard	RFC 6901 - Woot!

JSON Query

The screenshot shows a web browser window with the address bar displaying `https://www.sitepen.com/blog/2008/07/16/jsonquery-data-querying-beyond-jsonpath/`. The sitepen logo is in the top left, and navigation links for Services, Blog, About Us, Contact, and Login are in the top right. The main heading of the page is "JSONQuery: Data Querying Beyond JSONPath".

Notice: We recommend reading our newer post on [RQL](#).

A new data querying tool for has been added to [Dojo 1.2](#). JSONQuery is a new module intended to succeed and improve upon the JSONPath module introduced in Dojo 1.1. JSONQuery provides a comprehensive set of data querying tools including filtering, recursive search, sorting, mapping, range selection, and flexible expressions with wildcard string comparisons and various operators.

JSONQuery provides [safe evaluation with language agnostic expressions](#) that prevents arbitrary code execution. It also uses intuitive [result-based evaluation](#) that allows successive query operations. Furthermore, the new JSONQuery module provides significant performance improvements, with 20-100x faster execution with the common filter operation on large arrays than the JSONPath module. JSONQuery generally supersedes the functionality of JSONPath and provides syntax that matches and behaves like JavaScript where the syntax intersects for maximum ease of use.

Usage API

A JSONQuery can be executed with the following call:

Featured articles

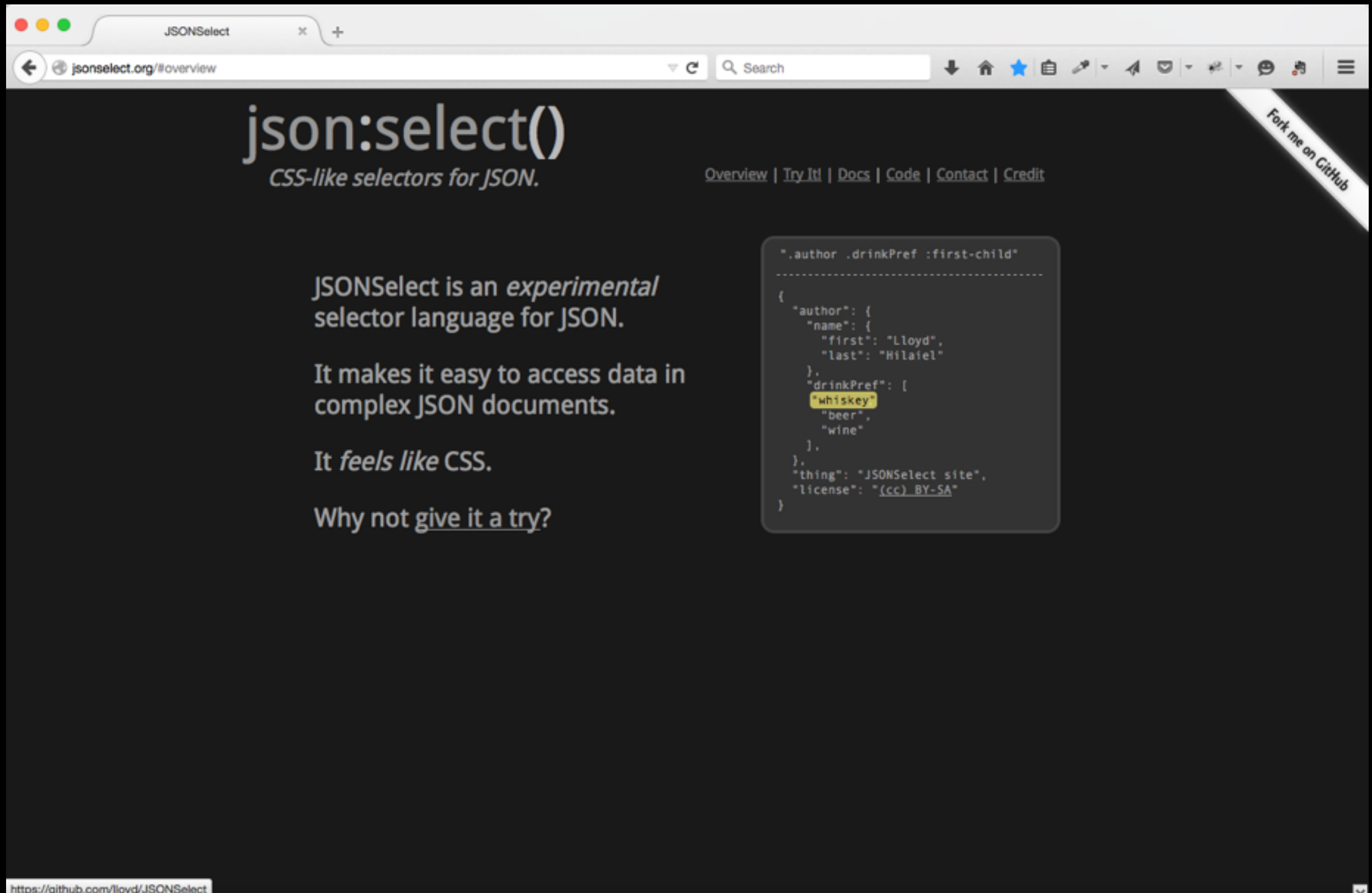
- Intern 2.2 released
- Introducing dstore
- On YUI, Dojo 2, and long-term JavaScript toolkits
- Testable code best practices
- Performance Comparison: dgrid OnDemandGrid and Dojo Grid

We're hiring!

JSON Query Scorecard

Mind Share	N?
Dev Community	N?
Platforms	JavaScript, Node.js
Intuitive	Y
Standard	N

json:select



The image is a screenshot of a web browser displaying the 'json:select.org' website. The browser's address bar shows 'jsonselect.org/#overview'. The website has a dark theme. At the top, the title 'json:select()' is displayed in a large, light-colored font, with the subtitle 'CSS-like selectors for JSON.' below it. To the right of the title, there are navigation links: 'Overview', 'Try It!', 'Docs', 'Code', 'Contact', and 'Credit'. A diagonal banner on the right side of the page says 'Fork me on GitHub'. The main content area on the left contains three paragraphs: 'JSONSelect is an *experimental* selector language for JSON.', 'It makes it easy to access data in complex JSON documents.', and 'It *feels like* CSS.' Below these is the question 'Why not give it a try?'. On the right side of the main content, there is a code block showing a JSON selector '.author .drinkPref :first-child' followed by a dashed line and a JSON object. The JSON object has an 'author' field with a 'name' object (containing 'first' and 'last') and a 'drinkPref' array (containing 'whiskey', 'beer', and 'wine'). The 'whiskey' value in the array is highlighted in yellow. The JSON object also has 'thing' and 'license' fields.

JSONSelect is an *experimental* selector language for JSON.

It makes it easy to access data in complex JSON documents.

It *feels like* CSS.

Why not give it a try?

```
.author .drinkPref :first-child
-----
{
  "author": {
    "name": {
      "first": "Lloyd",
      "last": "Hilael"
    },
    "drinkPref": [
      "whiskey",
      "beer",
      "wine"
    ],
    "thing": "JSONSelect site",
    "license": "(cc) BY-SA"
  }
}
```

<https://github.com/lloyd/JSONSelect>

json:select Expression Tester

The screenshot shows a web browser window with the title "JSONSelect Expression Tester". The address bar shows the URL "jsonselect.curiousconcept.com/#". The page has a teal header with the "JSONSELECT" logo and navigation links: "About", "Learn", "Changelog", and "Contact".

The main content area has a teal background. On the left, a white curved arrow points to a large white text input box labeled "JSON Data/URL". Below this box is another white text input box labeled "JSONSelect Expression". To the right of the "JSON Data/URL" box is a "JSON Template" dropdown menu currently set to "3 Space Tab". A white curved arrow points from the dropdown menu back towards the "JSON Data/URL" box.

Below the "JSONSelect Expression" input box is a teal button labeled "Process".

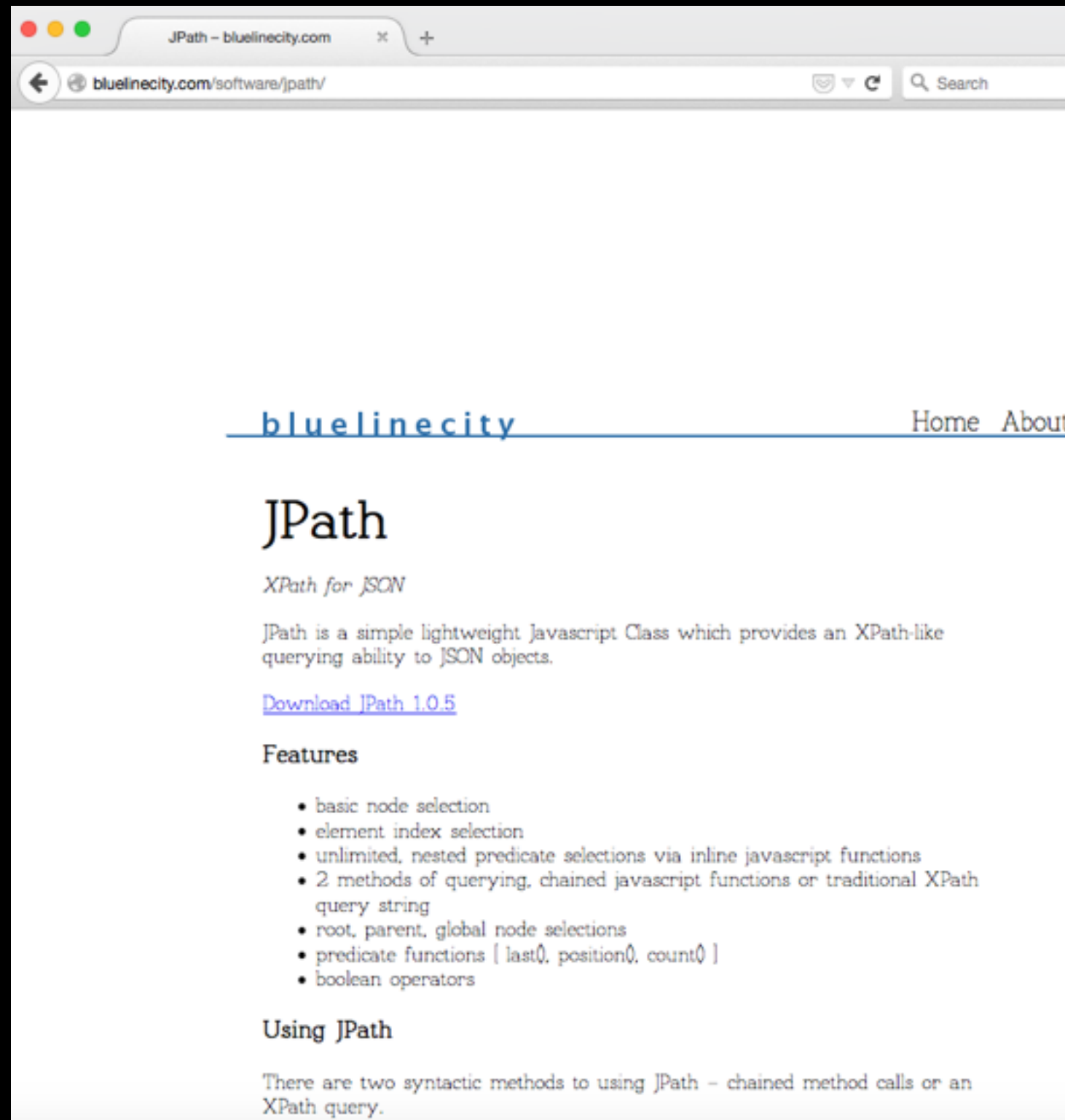
Text on the left side of the interface: "Paste in JSON or a URL, enter the JSONSelect and away you go."

json:select Demo

json:select Scorecard

Mind Share	Y?
Dev Community	Y?
Platforms	JavaScript, Node.js, Ruby
Intuitive	Y - CSS
Standard	N

JPath



blueinecity

Home About

JPath

XPath for JSON

JPath is a simple lightweight Javascript Class which provides an XPath-like querying ability to JSON objects.

[Download JPath 1.0.5](#)

Features

- basic node selection
- element index selection
- unlimited, nested predicate selections via inline javascript functions
- 2 methods of querying, chained javascript functions or traditional XPath query string
- root, parent, global node selections
- predicate functions [last(), position(), count()]
- boolean operators

Using JPath

There are two syntactic methods to using JPath – chained method calls or an XPath query.

JPath Scorecard

Mind Share	N
Dev Community	Y?
Platforms	JavaScript, Node.js, Ruby
Intuitive	Y
Standard	N

Jaql

The screenshot shows the Google Code project page for Jaql. The browser's address bar displays the URL `https://code.google.com/p/jaql/`. The page header includes the Jaql logo (a blue house with a curly brace and 'P' inside) and the text "jaql Query Language for JavaScript(r) Object Notation (JSON)". Navigation links for "Project Home", "Downloads", "Wiki", "Issues", "Source", and "Export to GitHub" are present. A search bar is also visible. The main content area is divided into a left sidebar and a main text area. The sidebar contains sections for "Project Information" (with links to "Project feeds", "Code license" (Apache License 2.0), and "Labels" (json, jaql, query, semi-structured, mapreduce, optimization, parallel)), "Members" (listing vik.erce...@gmail.com, Kevin.Be...@gmail.com, and 9 committers), "Featured" (with a "Downloads" section listing jaql-0.5.1_12_07_2010.tgz and a "Show all »" link), and "Links" (with a "Groups" section listing Jaql Users and Jaql Developers). The main text area contains a description of Jaql as a query language for JSON, its core features, and an announcement about its integration with IBM's BigInsights product.

jaql - Query Language for J... x +

https://code.google.com/p/jaql/ Search

My favorites | Sign in

jaql
Query Language for JavaScript(r) Object Notation (JSON)

Project Home Downloads Wiki Issues Source Export to GitHub

Summary People

Project Information

[Project feeds](#)

Code license
[Apache License 2.0](#)

Labels
json, jaql, query, semi-structured, mapreduce, optimization, parallel

Members
[vik.erce...@gmail.com](#),
[Kevin.Be...@gmail.com](#),
9 committers

Featured

Downloads
[jaql-0.5.1_12_07_2010.tgz](#)
[Show all »](#)

Links

Groups
[Jaql Users](#)
[Jaql Developers](#)

Jaql is a query language designed for Javascript Object Notation (JSON), a data format that has become popular because of its simplicity and modeling flexibility. Jaql is primarily used to analyze large-scale semi-structured data. Core features include user extensibility and parallelism. In addition to modeling semi-structured data, JSON simplifies extensibility. Hadoop's Map-Reduce is used for parallelism.

You can get started with the [JaqlOverview](#) of the current version, and check out the [RoadMap](#) of future changes.

Announcement: As you may have noticed, development here has been put on hold. The good news is that Jaql is a part of IBM's [BigInsights](#) product (updated [docs](#)) and we are very excited to help solve a very wide range of problems! A free download of [BigInsights](#) is available where you can try out Jaql (and some new features) along with the many other features in the product. It has been a hectic period for us, but we also want to let you know that we intend to and are working towards reviving Jaql's open source development as well. Please stay tuned and thank you for all of the feedback!

[Terms](#) - [Privacy](#) - [Project Hosting Help](#)

Powered by [Google Project Hosting](#)

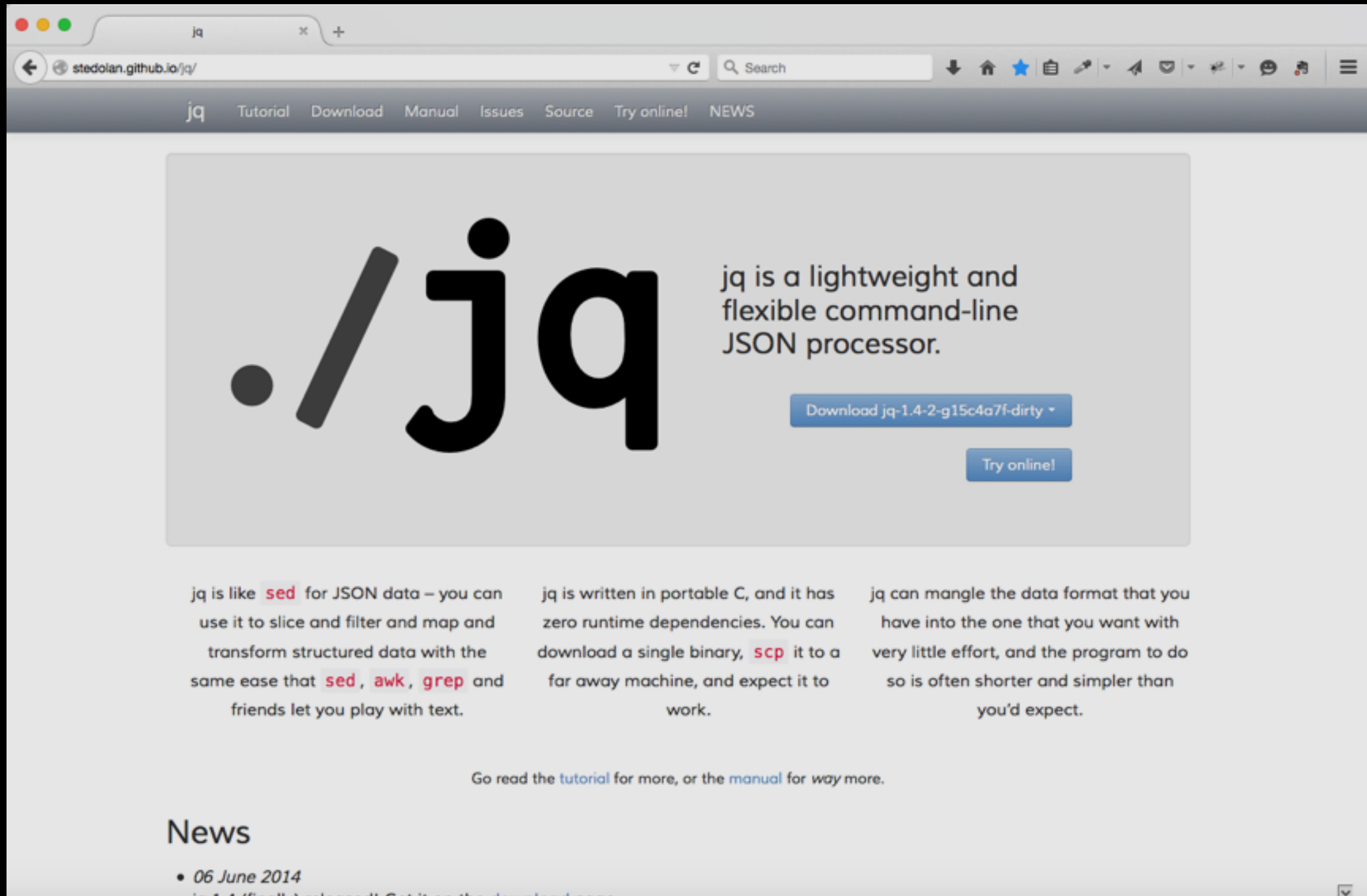
Jaql Example

```
1
2 Arrays
3 -----
4 jaql> a = [5, 6, 7];
5 jaql> a[1];    // 6
6 jaql> a[1:2];  // [6,7]
7
8
9 Objects
10 -----
11 jaql> a = { name : "fred", age : 55, children : ["tom", "anna"] };
12 jaql> a.name;    // "fred"
13 jaql> a.children[0]; // "tom"
14
```

Jaql Scorecard

Mind Share	N
Dev Community	N?
Platforms	JaqlShell, Hadoop
Intuitive	Y
Standard	N

jq



The screenshot shows the homepage of the jq website. At the top, there's a navigation bar with links: 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, it says 'jq is a lightweight and flexible command-line JSON processor.' Below this, there are two buttons: 'Download jq-1.4-2-g15c4a7f-dirty' and 'Try online!'. Further down, there are three columns of text describing jq's capabilities and ease of use. At the bottom, there's a 'News' section with a bullet point dated '06 June 2014' mentioning the release of jq 1.4.

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

The screenshot shows the jqplay.org website in a web browser. The browser's address bar displays "https://jqplay.org". The website's header includes the "jqplay" logo and the tagline "A playground for jq 1.4".

The main interface is divided into three primary sections:

- Filter:** A text input field at the top left, currently empty.
- JSON:** A large text area below the Filter section, containing a single line with the number "1".
- Result:** A large text area on the right side, also containing a single line with the number "1". Above this area are five checkboxes: "Compact Output", "Null Input", "Raw Input", "Raw Output", and "Slurp", all of which are currently unchecked.

At the bottom of the page is a "Cheatsheet" section. It includes a note: "Click on the icons ([icon]) in the table below to see examples." Below this note is a table with four rows, each showing a jq filter, its description, and a small icon.

<code>.</code>	unchanged input	[icon]	<code>,</code>	feed input into multiple filters	[icon]
<code>.foo, .foo.bar, .foo?</code>	value at key	[icon]	<code> </code>	pipe output of one filter to the next filter	[icon]

jq Examples

```
1
2 In jq-play
3 -----
4 .airports
5
6 .airports[10]
7
8 .airports[10] | { id, IATA, weather }
9
10 .airports[10:15] | .[] | { id, IATA, weather }
11
12
13 In curl
14 -----
15 curl 'http://localhost:5000/airports'
16
17 curl 'http://localhost:5000/airports' | jq .[10]
18
19 curl 'http://localhost:5000/airports' | jq '.[10] | { id, IATA, weather }'
20
21 curl 'http://localhost:5000/airports' | jq '.[10:15] | .[] | { id, IATA, weather }'
22
```

jq Scorecard

Mind Share	Y
Dev Community	Y
Platforms	CLI - Linux / Mac OS X / Windows
Intuitive	Y
Standard	N

My JSON Search Choices

API / Product	Type	Rank
JSON Pointer	API	1
JSON Path	API	2
json:select	API	3
JPath	API	3
JSON Query	API	4

My JSON Search Choices

API/Product	Type	Rank
jq	CLI	1

My JSON Search Choices

API/Product	Type	Rank
Jaql	Shell	1
JSONiq	Shell	2

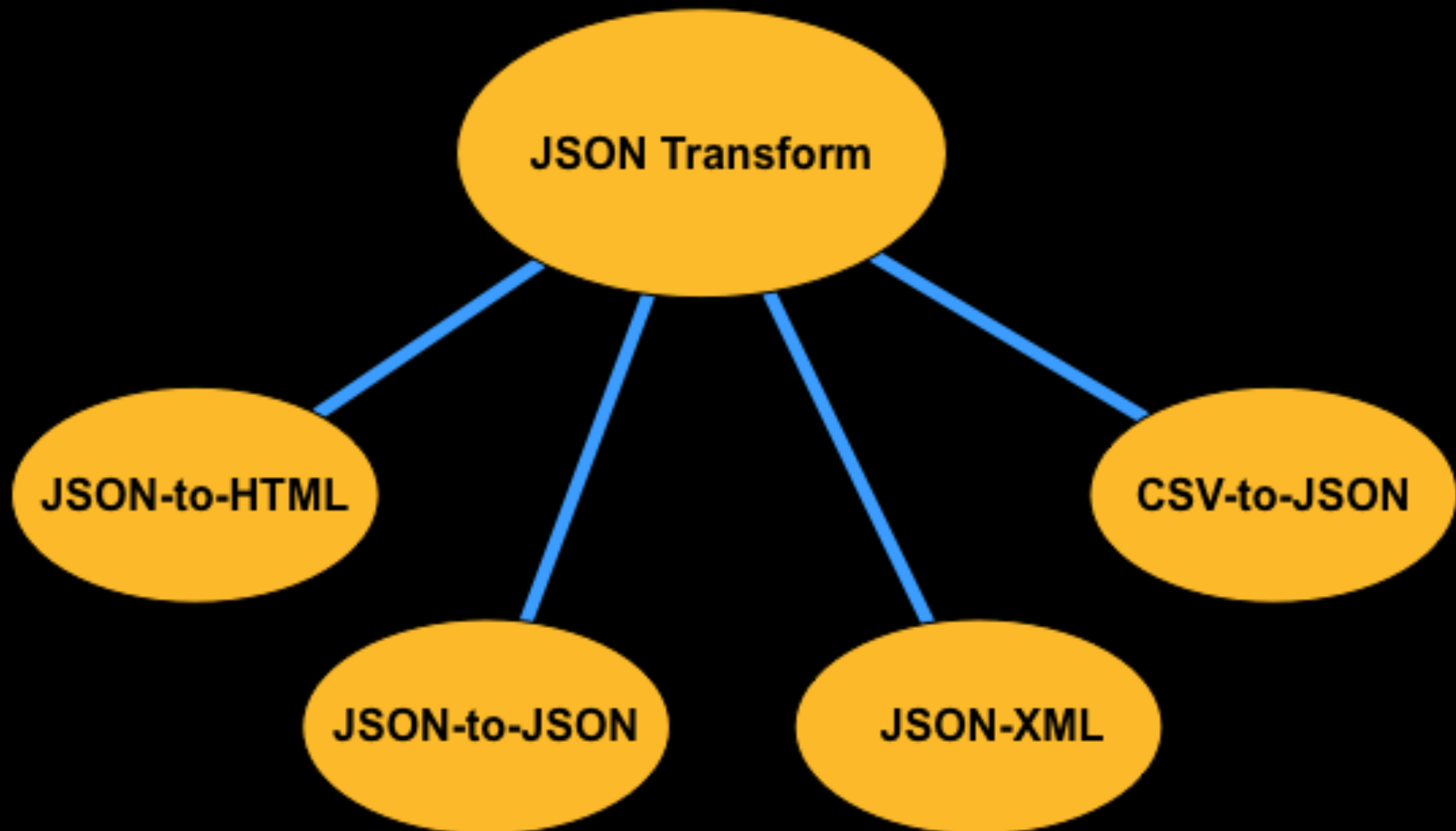
Where Are We?

Overview

JSON Search

JSON Transform

JSON Transform



JSON-T

The screenshot shows a web browser window with the address bar displaying 'goessner.net/articles/jsont/'. The page title is '<stefan.goessner/> Mechanik, das Web und der ganze Rest'. The navigation bar includes links for 'Home', 'Lehre', 'Download', and 'Info'. The main content area is titled '# Transforming JSON' with a date of '2006-01-30' and a version 'e1'. The text describes JSON as a lightweight text format for data interchange, better suited for structured data than XML. It mentions a frequently requested task is transforming JSON data into other formats like XML or HTML. The most obvious way to achieve this is using a programming language (ECMAScript, Ruby, etc.) and the DOM-API. In XML, documents can be transformed by another XML document containing transformation rules (XSLT) and applying these rules using an XSLT-processor. Adopting that concept, the author has been experimenting with a set of transformation rules (written in JSON). As a result, in analogy to XML/XSLT, the combination JSON/JSONT can be used to transform JSON data into any other format by applying a specific set of rules.

The second section is '# Introducing JSONT' with a date of '2006-01-30' and a version 'e2'. It starts with 'Let's start with a simple JSON object' and shows a JSON object:

```
{ "link": { "uri": "http://company.com", "title": "company homepage" } }
```

 which is to be transformed into an HTML link element:

```
<a href="http://company.com">company homepage</a>
```

 For doing this, a corresponding rule is written:

```
{ "link": "<a href=\"{link.uri}\">{link.title}</a>" }
```

 and using a processor like `jsonT(data, rules)` the given rule can be applied to the data.

On the right side, there is a search bar with a 'Google Search' button and a 'Web' radio button. Below that is a table of contents titled '» Inhalt ..' listing links to 'Home', 'Lehre', 'Dynamik', 'Articles', 'DOM Events', 'Wiky', '2D Vectors', 'Slideous', 'JsonT', 'JSONPath', 'SVG', 'Download', 'Admin', and 'Info'. At the bottom right, there is a section titled '» comments ..' with a link to 'Exercise 09'.

JSON-T Example

simple array

```
["red", "green", "blue"]
```

+

```
["self": "<ul>\n{$}</ul>",  
 "self[*]": "  <li>{$}</li>\n"]
```

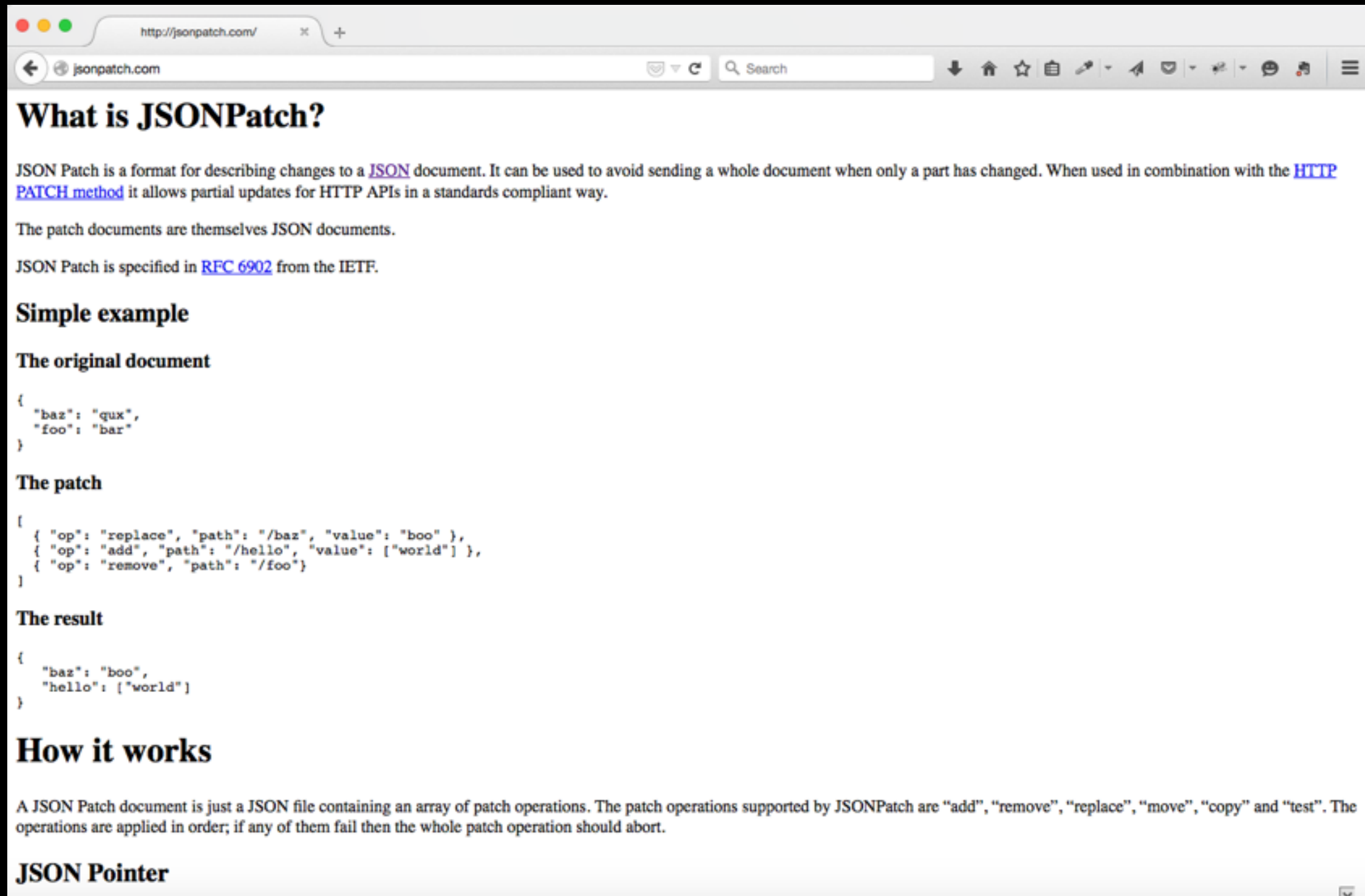
=

```
<ul>  
  <li>red</li>  
  <li>green</li>  
  <li>blue</li>  
</ul>
```

JSON-T Scorecard

Mind Share	N
Dev Community	N
Platforms	JavaScript, Node.js
Intuitive	Y
Standard	N

JSON Patch

A screenshot of a web browser displaying the jsonpatch.com website. The browser's address bar shows 'http://jsonpatch.com/'. The website has a clean, minimalist design with a white background and black text. The main heading is 'What is JSONPatch?'. Below it, there is a paragraph explaining that JSON Patch is a format for describing changes to a JSON document, used in combination with the HTTP PATCH method. It also mentions that patch documents are themselves JSON documents and that the format is specified in RFC 6902. The next section is 'Simple example', which contains three sub-sections: 'The original document', 'The patch', and 'The result', each followed by a JSON code block. The final section is 'How it works', which explains that a JSON Patch document is a JSON file containing an array of patch operations, and lists the supported operations: 'add', 'remove', 'replace', 'move', 'copy', and 'test'. The page ends with a 'JSON Pointer' section header. The browser's toolbar and various icons are visible at the top of the page.

What is JSONPatch?

JSON Patch is a format for describing changes to a [JSON](#) document. It can be used to avoid sending a whole document when only a part has changed. When used in combination with the [HTTP PATCH method](#) it allows partial updates for HTTP APIs in a standards compliant way.

The patch documents are themselves JSON documents.

JSON Patch is specified in [RFC 6902](#) from the IETF.

Simple example

The original document

```
{
  "baz": "qux",
  "foo": "bar"
}
```

The patch

```
[
  { "op": "replace", "path": "/baz", "value": "boo" },
  { "op": "add", "path": "/hello", "value": ["world"] },
  { "op": "remove", "path": "/foo" }
]
```

The result

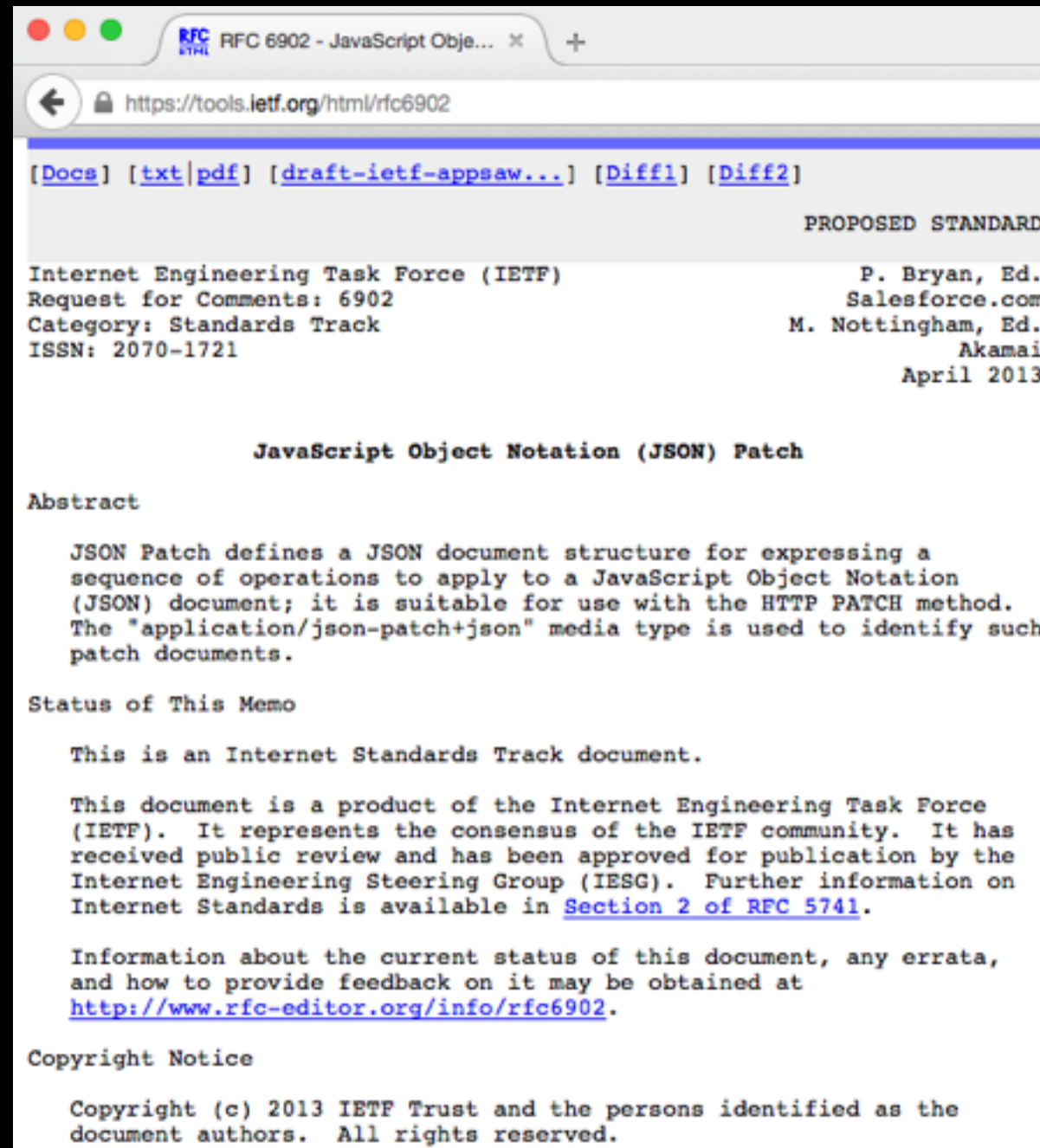
```
{
  "baz": "boo",
  "hello": ["world"]
}
```

How it works

A JSON Patch document is just a JSON file containing an array of patch operations. The patch operations supported by JSONPatch are "add", "remove", "replace", "move", "copy" and "test". The operations are applied in order; if any of them fail then the whole patch operation should abort.

JSON Pointer

JSON Patch Standard



The screenshot shows a web browser window with the title "RFC 6902 - JavaScript Object Notation (JSON) Patch". The address bar shows the URL "https://tools.ietf.org/html/rfc6902". The page content includes links for [Docs], [txt|pdf], [draft-ietf-appsaw...], [Diff1], and [Diff2]. The status is "PROPOSED STANDARD". The authors listed are P. Bryan, Ed. (Salesforce.com) and M. Nottingham, Ed. (Akamai), dated April 2013. The title "JavaScript Object Notation (JSON) Patch" is centered. The abstract describes the JSON Patch document structure for expressing a sequence of operations to apply to a JSON document. The status of the memo is "This is an Internet Standards Track document." The document is a product of the IETF, representing the consensus of the IETF community, approved for publication by the IESG. Information about the current status, errata, and feedback is available at <http://www.rfc-editor.org/info/rfc6902>. The copyright notice states: Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

[Docs] [txt|pdf] [draft-ietf-appsaw...] [Diff1] [Diff2]

PROPOSED STANDARD

Internet Engineering Task Force (IETF)
Request for Comments: 6902
Category: Standards Track
ISSN: 2070-1721

P. Bryan, Ed.
Salesforce.com
M. Nottingham, Ed.
Akamai
April 2013

JavaScript Object Notation (JSON) Patch

Abstract

JSON Patch defines a JSON document structure for expressing a sequence of operations to apply to a JavaScript Object Notation (JSON) document; it is suitable for use with the HTTP PATCH method. The "application/json-patch+json" media type is used to identify such patch documents.

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/rfc6902>.

Copyright Notice

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

JSON Patch Example

Simple example

The original document

```
{  
  "baz": "gux",  
  "foo": "bar"  
}
```

The patch

```
[  
  { "op": "replace", "path": "/baz", "value": "boo" },  
  { "op": "add", "path": "/hello", "value": ["world"] },  
  { "op": "remove", "path": "/foo"}  
]
```

The result

```
{  
  "baz": "boo",  
  "hello": ["world"]  
}
```

JSON Patch Scorecard

Mind Share	Y?
Dev Community	Y
Platforms	JavaScript, Node.js, Java, Ruby, etc.
Intuitive	Y
Standard	RFC 6902 - Yes!

My JSON Transform Choices

API / Product	Type	Rank
JSON Patch	JSON-to-JSON	1
JSON-T	JSON-to-JSON	2

Our Agenda

Overview

JSON Search

JSON Transform

What's The Point?

JSON Search and Transform ...

Simplify interaction with RESTful APIs

Questions?

Tom Marris

thomasamarris@comcast.net
@TomMarris



JSON Resources

JSON Spec - <http://tools.ietf.org/html/rfc4627>

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

JSONLint - <http://www.jsonlint.com>

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

JSON Resources

JSONiq - <http://www.jsoniq.org/>

JSOINT - <http://goessner.net/articles/jsont/>

JSONPath Resources

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

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

<https://rubygems.org/gems/jsonpath/versions/0.5.6>

<https://www.npmjs.com/package/json-path>

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

JSON Pointer Resources

<https://tools.ietf.org/html/rfc6901>

<https://www.npmjs.com/package/json-pointer>

<https://rubygems.org/gems/json-pointer>

<https://github.com/fge/jackson-coreutils>

<http://susanpotter.net/blogs/software/2011/07/why-json-pointer-falls-short/>

<https://zato.io/blog/posts/json-pointer-rfc-6901.html>

JSON Query Resources

<https://github.com/jcrosby/jsonquery>

<https://github.com/mmckegg/json-query>

<https://www.npmjs.com/package/json-query>

json:select Resources

<http://jsonselect.org/#overview>

<https://github.com/lloyd/JSONSelect>

<https://github.com/lloyd/JSONSelect/blob/master/JSONSelect.md>

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

https://github.com/fd/json_select

JPath Resources

<http://bluelinecity.com/software/jpath/>

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

<https://www.npmjs.com/package/node-jpath>

<https://github.com/merimond/jpath>

jaql Resources

<https://code.google.com/p/jaql/>

<http://www.havlena.net/en/programming/jaql-in-hadoop-a-brief-introduction/>

http://www-01.ibm.com/support/knowledgecenter/SSPT3X_3.0.0/com.ibm.swg.im.infosphere.biginsights.jaql.doc/doc/c0057749.html

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-T Resources

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

<https://github.com/CamShaft/jsont>

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

JSON Patch Resources

<http://jsonpatch.com/>

<https://tools.ietf.org/html/rfc6902>

<http://jsonpatchjs.com/>

https://rubygems.org/gems/json_patch

<https://github.com/flipkart-incubator/zjsonpatch>

<https://www.npmjs.com/package/json-patch>

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

JSON Groups

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

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