

# JSON at Work: Schema

**Tom Marrs**

**@TomMarrs**



# About Me ...



# What's The Point?

## Drive API Design with JSON Schema

# Our Agenda

**JSON Schema Overview**

1

**Core JSON Schema**

2

**API Design with JSON Schema**

3

# Your Takeaway

Core JSON Schema + JSON Workflow

# We're Not Covering

REST

Deep JS

Other Languages

# Examples and Slides

[https://github.com/tmarrs/presentations/tree/master/  
JSON-at-Work-Schema](https://github.com/tmarrs/presentations/tree/master/JSON-at-Work-Schema)

# Where Are We?

JSON Schema Overview

1

Core JSON Schema

2

API Design with JSON Schema

3

# What is JSON Schema?

**Validate Structure + Format**

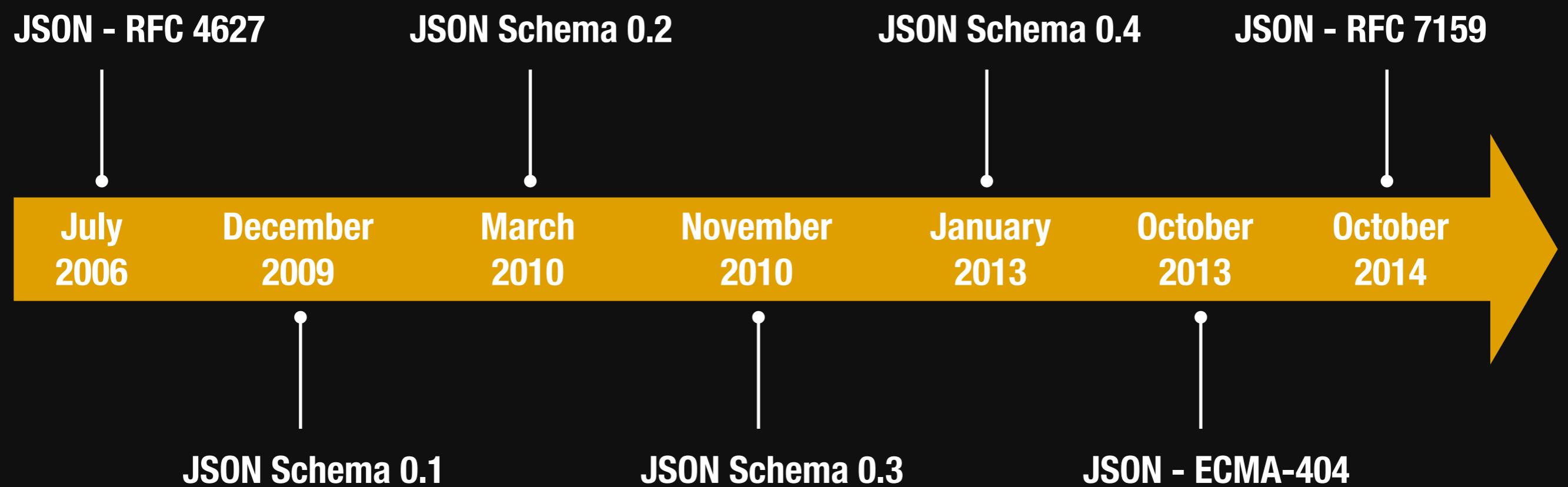
# Basic JSON Schema

```
1
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "email": {
7              "type": "string"
8          },
9          "firstName": {
10             "type": "string"
11         },
12         "lastName": {
13             "type": "string"
14         }
15     }
16 }
```

# Basic JSON - Document

```
2  {
3      "email": "larsonrichard@ecratic.com",
4      "firstName": "Larson",
5      "lastName": "Richard"
6 }
```

# JSON Schema Timeline - When?



# The Journey ...



The screenshot shows the homepage of json-schema.org as it would appear in a desktop web browser. The page has a dark green header bar with the title "json-schema.org" and the subtitle "The home of JSON Schema". Below the header is a navigation menu with four items: "about", "docs", "examples", and "software", where "about" is underlined to indicate it is the current section. The main content area is divided into several horizontal sections by thin black lines. The first section is titled "What does it do?" and contains two bullet points: "JSON Schema describes your JSON data format" and "JSON Hyper-Schema turns your JSON data into hyper-text". The second section is titled "Advantages" and lists the benefits of both JSON Schema and JSON Hyper-Schema. The third section is titled "More" and includes a link to check out the specification and examples. The browser interface at the top includes standard elements like tabs, a search bar, and a toolbar.

JSON Schema describes your JSON data format

JSON Hyper-Schema turns your JSON data into hyper-text

**Advantages**

**JSON Schema**

- describes your existing data format
- clear, human- and machine-readable documentation
- complete structural validation, useful for
  - automated testing
  - validating client-submitted data

**JSON Hyper-Schema**

- describes your existing API - no new structures required
- links (including [URI Templates](#) for target URLs)
- forms - specify a JSON Schema for the desired data

**More**

Interested? Check out:

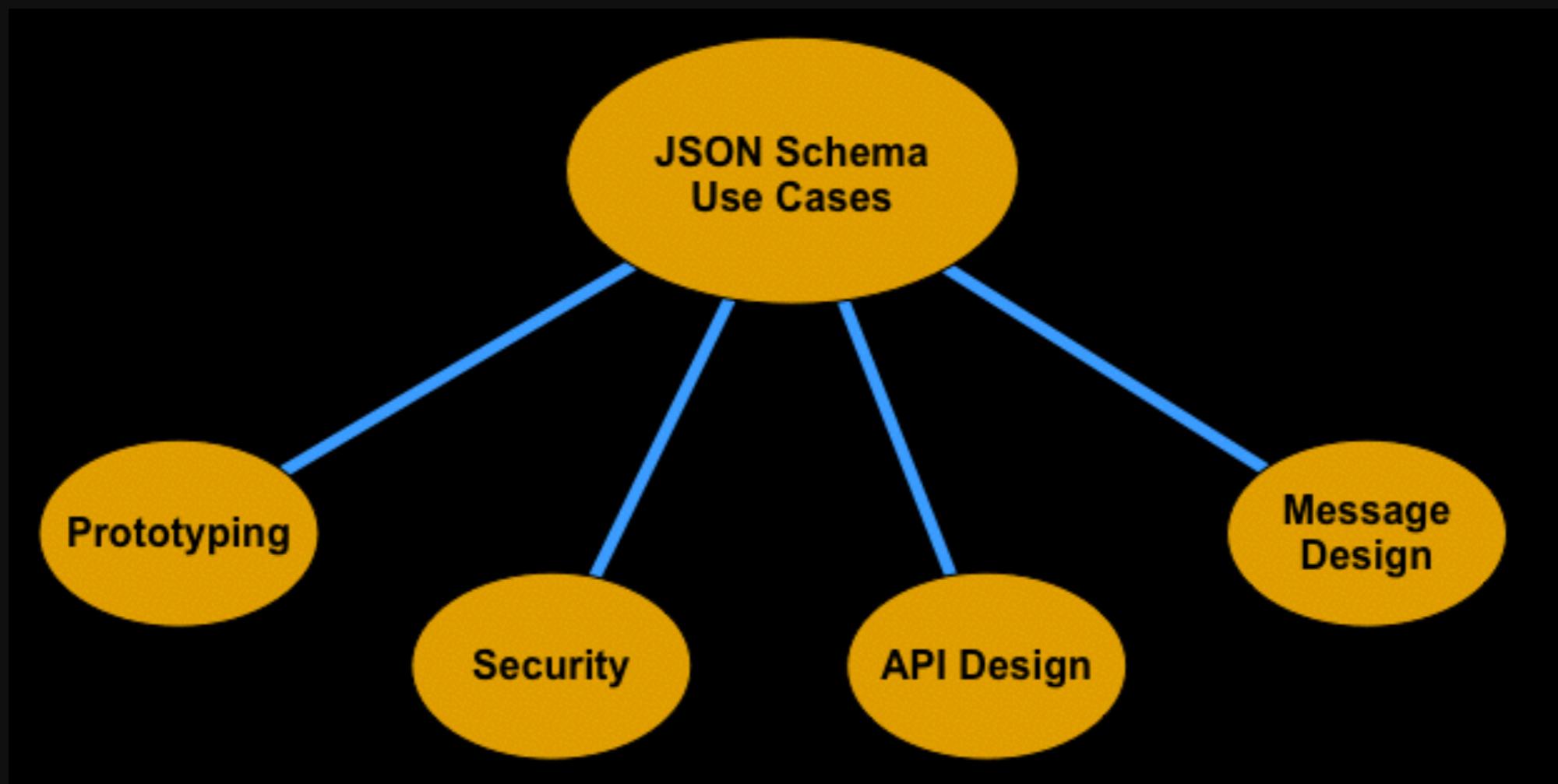
- the [specification](#)
- some [examples](#)

# JSON Schema on GitHub

The screenshot shows a GitHub repository page for 'kriszyp/json-schema'. The page header includes the repository name, a search bar, and navigation links. Below the header, a summary bar displays statistics: 67 commits, 2 branches, 2 releases, and 7 contributors. The main content area shows a list of commits, each with a file icon, author (kriszyp), date (authored on Aug 30, 2012), commit message, and time ago. The commits are listed in reverse chronological order.

File	Commit Message	Time Ago
draft-00	Schema URIs are now namespace versioned.	5 years ago
draft-01	Schema URIs are now namespace versioned.	5 years ago
draft-02	Schema URIs are now namespace versioned.	5 years ago
draft-03	Fix hyper-schema syntax error.	4 years ago
draft-04	Core changes:	4 years ago
lib	Clean out modifications to primitives	4 years ago
test	Add vows-based unit tests.	4 years ago
CNAME	Add CNAME, I think this is needed to have json-schema.org point to it.	3 years ago
README.md	Updated docs	5 years ago
draft-zyp-json-schema-03.xml	Merge http://github.com/garycourt/json-schema	4 years ago
draft-zyp-json-schema-04.xml	Merge git://github.com/garycourt/json-schema	4 years ago
package.json	bump version	4 years ago

# Where Does JSON Schema Fit?



# Who uses JSON Schema?



Swagger



# Why isn't JSON Validation Enough?

**Semantics**

**Schema + Instance  
Document**

**Meaning**

**Ex: Person, Order**

**Structure**

**Instance Document**

**Well-formed - Valid JSON**

**Syntax**

# Haven't We Seen This Before?

**JSON Schema**

**XML Schema**

**No Reference**

**Instance Document  
references Schema**

**No Namespace - Yes!**

**Namespace Misery**

**.json**

**.xsd**

# Where Are We?

**JSON Schema Overview**

1

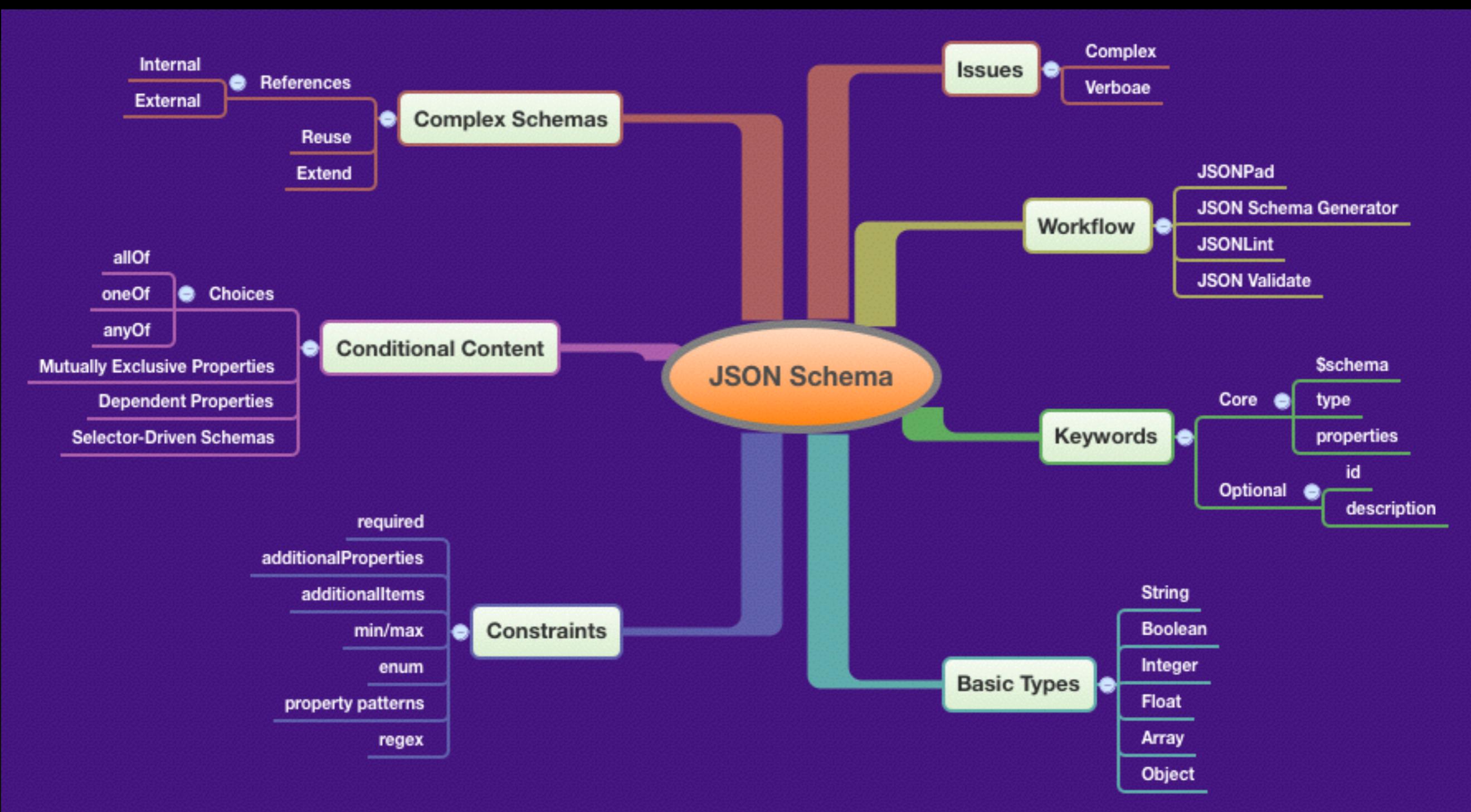
**Core JSON Schema**

2

**API Design with JSON Schema**

3

# Facets of JSON Schema



# My JSON Schema Workflow

**Model JSON Document**

JSONPad

[https://  
code.google.com/p/  
json-pad/](https://code.google.com/p/json-pad/)

**Generate JSON Document**

JSON Generator

<http://jsonschema.net/>

**Validate JSON Document**

JSON Validate

<http://jsonvalidate.com/>

JSONLint

<http://jsonlint.com>

# JSONLint

The screenshot shows a web browser window for JSONLint. The address bar says "JSONLint - The JSON Valid..." and the URL is "jsonlint.com". The page content includes a code editor with JSON code, validation buttons, and a results section.

Want more from JSONLint? Try [JSONLint Pro](#)

Props to Douglas Crockford of [JSON](#) and [JS Lint](#) and Zach Carter, who provided the pure JS implementation of jsonlint.

```
1 | {
2 |   "email": "larsonrichard@eclaric.com",
3 |   "firstName": "Larson",
4 |   "lastName": "Richard"
5 |
6 |
7 |
8 |
9 |
10|
11|
12|
13|
14|
15|
16|
17|
18|
19|
20|
21| }
```

**Validate** [FAQ](#)

JSON Lint is an idea from Arc90's Kindling

**Kindling**

**Results**

Valid JSON

# JSON Validate

The screenshot shows the JSON Validate interface on jsonvalidate.com. The top navigation bar includes tabs for 'Import', 'About', and 'Help'. The main area is divided into two code editors: 'JSON Schema' on the left and 'JSON Content' on the right. The 'JSON Schema' editor contains the following JSON:

```
1 {
2   "$schema": "http://json-schema.org/draft-04/schema#",
3   "type": "object",
4   "properties": {
5     "email": {
6       "type": "string"
7     },
8     "firstName": {
9       "type": "string"
10    },
11    "lastName": {
12      "type": "string"
13    }
14  }
15
16
17
18
19
20
21
^~
```

The 'JSON Content' editor contains the following JSON, with the 'lastName' field underlined in red, indicating a validation error:

```
1 {
2   "email": "larsonrichard@ecratic.com",
3   "firstName": "Larson",
4   "lastName": "Richard"
5 }
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
^~
```

Below the code editors are sections for 'References' and 'Results'. The 'References' section shows a numbered list from 1 to 8. The 'Results' section displays the word 'Valid'.

At the bottom, there are buttons for 'Validate' and 'Reset all', and a link to 'Learn more about Using JSON Schema' with a 'UJS' logo.

Validate    Reset all    Learn more about Using JSON Schema    UJS

# Beware of Wonky WiFi - Hedge Your Bets!



PDD - Presentation-Driven Development

# jsonlint

```
npm install -g jsonlint
```

```
jsonlint basic.json
```

<https://github.com/zaach/jsonlint>

# ujs-jsonvalidate

```
npm install -g ujs-jsonvalidate
```

```
validate basic.json basic-schema.json
```

<https://github.com/usingjsonschema/ujs-jsonvalidate-nodejs>

# Basic Keywords

Keyword	Definition
\$schema	<p>Specify JSON Schema version -</p> <pre data-bbox="912 774 2535 970">"schema": "<u>http://json-schema.org/draft-04/schema#</u>"</pre>
type	<p>The data type -</p> <pre data-bbox="912 1204 1841 1298">"type": "string"</pre>
properties	The fields for an object

# Optional Keywords

Keyword	Definition
id	(1): Path to field (2): URL to Schema
description	For documentation

# Basic Types - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "email": {
7              "type": "string"
8          },
9          "firstName": {
10             "type": "string"
11         },
12         "lastName": {
13             "type": "string"
14         },
15         "age": {
16             "type": "integer"
17         },
18         "postedSlides": {
19             "type": "boolean"
20         },
21         "rating": {
22             "type": "number"
23         }
24     }
25 }
```

# Basic Types - JSON Document

```
1  {
2      "email": "larsonrichard@ecratic.com",
3      "firstName": "Larson",
4      "lastName": "Richard",
5      "age": 39,
6      "postedSlides": true,
7      "rating": 4.1
8  }
```

# Where's the Validation?

Keyword	Definition
<code>additionalProperties</code>	enable/disable additional fields in an object
<code>required</code>	Which fields are required
<code>additionalItems</code>	enable/disable additional array elements

# Basic Types Validation - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "email": {
7              "type": "string"
8          },
9          "firstName": {
10             "type": "string"
11         },
12         "lastName": {
13             "type": "string"
14         },
15         "postedSlides": {
16             "type": "boolean"
17         },
18         "rating": {
19             "type": "number"
20         }
21     },
22     "additionalProperties": false
23 }
```

# Validation with Required - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "email": {
7              "type": "string"
8          },
9          "firstName": {
10             "type": "string"
11         },
12         "lastName": {
13             "type": "string"
14         },
15         "postedSlides": {
16             "type": "boolean"
17         },
18         "rating": {
19             "type": "number"
20         }
21     },
22     "additionalProperties": false,
23     "required": ["email", "firstName", "lastName", "postedSlides", "rating"]
24 }
```

# Validation with Required - JSON Doc

```
2  {
3      "email": "larsonrichard@ecratic.com",
4      "firstName": "Larson",
5      "lastName": "Richard",
6      "rating": 4.1
7  }
```

# Number min/max - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "rating": { "type": "number", "minimum": 1.0, "maximum": 5.0 }
7      },
8      "additionalProperties": false,
9      "required": ["rating"]
10 }
```

# Number min/max - JSON Doc

```
2  {
3    "rating": "4.2"
4 }
```

# Simple Array - JSON Schema

```
1
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "tags": {
7              "type": "array",
8              "items": {
9                  "type": "string"
10             },
11             "additionalItems": false
12         }
13     },
14     "additionalProperties": false,
15     "required": ["tags"]
16 }
```

# Simple Array - JSON Doc

```
2  {
3      "tags": ["fred"]
4 }
```

# Array min/max - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "tags": {
7              "type": "array",
8              "minItems": 2,
9              "maxItems": 4,
10             "items": {
11                 "type": "string"
12             },
13             "additionalItems": false
14         }
15     },
16     "additionalProperties": false,
17     "required": ["tags"]
18 }
```

# Array min/max - JSON Doc

```
2  {
3      "tags": ["fred", "a"]
4 }
```

# Array Enum - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "tags": {
7              "type": "array",
8              "minItems": 2,
9              "maxItems": 4,
10             "items": {
11                 "type": "string",
12                 "enum": [
13                     "Open Source", "Java", "JavaScript", "JSON", "REST"
14                 ]
15             },
16             "additionalItems": false
17         }
18     },
19     "additionalProperties": false,
20     "required": ["tags"]
21 }
```

# Array Enum - JSON Doc

```
2  {
3      "tags": ["Java", "REST"]
4 }
```

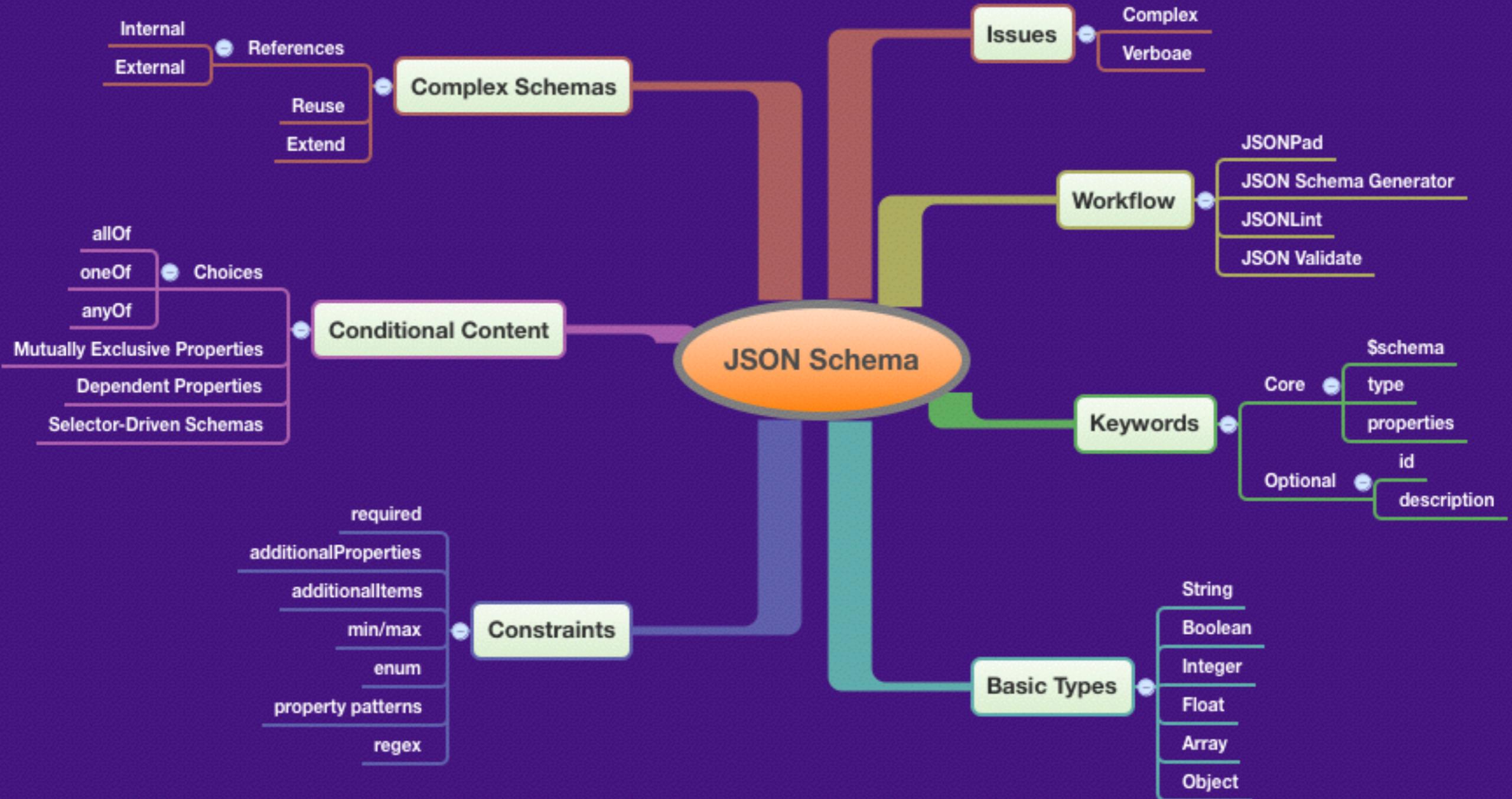
# Named Object - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "speaker": {
7              "type": "object",
8              "properties": {
9                  "firstName": { "type": "string" },
10                 "lastName": { "type": "string" },
11                 "email": { "type": "string" },
12                 "postedSlides": { "type": "boolean" },
13                 "rating": { "type": "number" },
14                 "tags": {
15                     "type": "array",
16                     "items": { "type": "string" },
17                     "additionalItems": false
18                 }
19             },
20             "additionalProperties": false,
21             "required": ["firstName", "lastName", "email",
22                         "postedSlides", "rating", "tags"
23                     ]
24         }
25     },
26     "additionalProperties": false,
27     "required": ["speaker"]
28 }
```

# Named Object - JSON Doc

```
2  {
3    "speaker": {
4      "firstName": "Larson",
5      "lastName": "Richard",
6      "email": "larsonrichard@ecratic.com",
7      "postedSlides": true,
8      "rating": 4.1,
9      "tags": [
10        "JavaScript", "AngularJS", "Yeoman"
11      ]
12    }
13  }
```

# Facets of JSON Schema



# Property Patterns - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "city": { "type": "string" },
7          "state": { "type": "string" },
8          "zip": { "type": "string" },
9          "country": { "type": "string" }
10     },
11     "patternProperties": {
12         "^line[1-3]$": { "type": "string" }
13     },
14     "additionalProperties": false,
15     "required": ["city", "state", "zip", "country"]
16 }
```

# Property Patterns - JSON Doc

```
2  {
3      "line1": "555 Main Street",
4      "line2": "#2",
5      "city": "Denver",
6      "state": "CO",
7      "zip": "80231",
8      "country": "USA"
9  }
```

# Regex - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "line1": { "type": "string" },
7          "city": { "type": "string" },
8          "state": { "type": "string" },
9          "zip": {
10             "type": "string",
11             "pattern": "^[0-9]{5}(-[0-9]{4})?$"
12         },
13         "country": { "type": "string" }
14     },
15     "additionalProperties": false,
16     "required": ["line1", "city", "state", "zip", "country"]
17 }
```

# Regex - JSON Doc

```
2  {
3      "line1": "555 Main Street",
4      "city": "Denver",
5      "state": "CO",
6      "zip": "80231",
7      "country": "USA"
8 }
```

# Help!! I'm Awful at Regex! - Regex101

The screenshot shows the Regex101 online regex tester and debugger interface. The main area is divided into several sections:

- REGULAR EXPRESSION:** A text input field containing the regex pattern `/insert your regular expression here/`. To its right is a status indicator `gmixXsuUAJ ? NO MATCH`.
- TEST STRING:** A text input field containing the test string `insert your test string here`.
- EXPLANATION:** A panel stating, "An explanation of your regex will be automatically generated as you type."
- MATCH INFORMATION:** A panel stating, "Detailed match information will be displayed here automatically."
- QUICK REFERENCE:** A table of tokens and their descriptions:

FULL REFERENCE	MOST USED TOKENS
most used token	A single character or... [abc]
all tokens	A character except... [^abc]
CATEGORIES	A character in the range [a-z]
general tokens	A character not in the range [^a-z]
anchors	

On the left sidebar, there are tabs for **SAVE & S...**, **FLAVOR** (selected), **PCRE**, **JS**, and **PY**. Below these are **TOOLS** with icons for clipboard, copy, paste, and checkmark.

# Regular Expressions Info

The screenshot shows the homepage of Regular-Expressions.info. At the top, there's a navigation bar with links for Quick Start, Tutorial, Tools & Languages, Examples, Reference, and Book Reviews. Below the navigation is a sidebar with links for Welcome, Regular Expressions Quick Start, Regular Expressions Tutorial, Replacement Strings Tutorial, Applications and Languages, Regular Expressions Examples, Regular Expressions Reference, Replacement Strings Reference, Book Reviews, Printable PDF, About This Site, and RSS Feed & Blog. A "Make a Donation" button is also present. The main content area features a section about RegexBuddy with a screenshot of the software interface showing a tree view of regex components. Below this is a welcome message and a detailed explanation of what regular expressions are and how they can be used. At the bottom, there's a book cover for 'Regular Expressions Cookbook' and a footer with a Runscope monitoring badge.

Regular-Expressions.info

Welcome

Regular Expressions Quick Start

Regular Expressions Tutorial

Replacement Strings Tutorial

Applications and Languages

Regular Expressions Examples

Regular Expressions Reference

Replacement Strings Reference

Book Reviews

Printable PDF

About This Site

RSS Feed & Blog

Easily create and understand regular expressions today.

Compose and analyze regex patterns with RegexBuddy's easy-to-grasp regex blocks and intuitive regex tree, instead of or in combination with the traditional regex syntax. Developed by the author of this website, RegexBuddy makes learning and using regular expressions easier than ever. [Get your own copy of RegexBuddy now](#)

**Welcome to Regular-Expressions.info**

**The Premier website about Regular Expressions**

A regular expression (regex or regexp for short) is a special text string for describing a search pattern. You can think of regular expressions as wildcards on steroids. You are probably familiar with wildcard notations such as \*.txt to find all text files in a file manager. The regex equivalent is .\*\.\.txt\$.

But you can do much more with regular expressions. In a text editor like [EditPad Pro](#) or a specialized text processing tool like [PowerGREP](#), you could use the regular expression \b[A-Z0-9.\_%+-]+@[A-Z0-9.-]+\.\.[A-Z]{2,4}\b to search for an email address. Any email address, to be exact. A very similar regular expression (replace the first \b with ^ and the last one with \$) can be used by a programmer to check whether the user entered a [properly formatted email address](#). In just one line of code, whether that code is written in [Perl](#), [PHP](#), [Java](#), [a .NET language](#), or a multitude of other languages.

It's going to be **200 OK** ● ©Runscope

# Regexr

The screenshot shows the RegExr v2.0 interface. On the left, there's a sidebar titled "Escaped characters" listing various escape sequences with their corresponding regex patterns. The main area has tabs for "Expression" and "Text". The "Expression" tab contains the regex pattern `/([A-Z])\w+/g`, which matches uppercase letters followed by one or more word characters. The "Text" tab displays the string "Welcome to RegExr v2.0 by gskinner.com!". Below the text, instructions for using the tool are provided. A "Sample text for testing" section lists several strings for regex matching.

Escaped characters

octal escape	\000
hexadecimal escape	\xFF
unicode escape	\uFFFF
control character escape	\cI
tab	\t
line feed	\n
vertical tab	\v
form feed	\f

Some characters have special meaning in regular expressions and must be escaped. All escaped characters begin with the \ character.

Within a character set, only \, -, and ] need to be escaped.

Expression

/([A-Z])\w+/g

16 matches

Text

Welcome to RegExr v2.0 by gskinner.com!

Edit the Expression & Text to see matches. Roll over matches or the expression for details. Undo mistakes with cmd-z. Save & Share expressions with friends or the Community. A full Reference & Help is available in the Library, or watch the video Tutorial.

Sample text for testing:

abcdefghijklmnopqrstuvwxyz ABCDEFGHIJKLMNOPQRSTUVWXYZ  
0123456789 +-.,!@#\$%^&\*();\//|<>''  
12345 -98.7 3.141 .6180 9,000 +42  
555.123.4567 +1-(800)-555-2468  
foo@demo.net bar.ba@test.co.uk  
www.demo.com http://foo.co.uk/  
http://regexr.com/foo.html?q=bar

Substitution

# Dependent Properties - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "businessName": { "type": "string" },
7          "contactName": { "type": "string" },
8          "line1": { "type": "string" },
9          "city": { "type": "string" },
10         "state": { "type": "string" },
11         "zip": { "type": "string" },
12         "country": { "type": "string" }
13     },
14     "additionalProperties": false,
15     "required": ["line1", "city", "state", "zip", "country"],
16     "dependencies": {
17         "contactName": ["businessName"]
18     }
19 }
```

# Dependent Properties - JSON Doc

```
2  {
3    "businessName": "My Conference, Inc.",
4    "contactName": "Larson Richard",
5    "line1": "555 Main Street",
6    "city": "Denver",
7    "state": "CO",
8    "zip": "80231",
9    "country": "USA"
10 }
```

# References (Internal) - JSON Schema

```
2  {
3      "$schema": "http://json-schema.org/draft-04/schema#",
4      "type": "object",
5      "properties": {
6          "line1": { "type": "string" },
7          "city": { "type": "string" },
8          "state": { "type": "string" },
9          "zip": { "$ref": "#/definitions/US_zipCode" },
10         "country": { "type": "string" }
11     },
12     "additionalProperties": false,
13     "required": ["line1", "city", "state", "zip", "country"],
14
15     "definitions": {
16         "US_zipCode": {
17             "type": "string",
18             "pattern": "^[0-9]{5}(-[0-9]{4})?$"
19         }
20     }
21 }
```

# References (Internal) - JSON Doc

```
2  {
3      "line1": "555 Main Street",
4      "city": "Denver",
5      "state": "CO",
6      "zip": "80231",
7      "country": "USA"
8  }
```

# References (External) - JSON Schema

**ex-13-external-ref-schema.json**

# References (Reused) - JSON Schema

**ex-13-USCommonAddrSchema.json**

# References (External) - JSON Doc

```
2  {
3    "line1": "555 Main Street",
4    "city": "Denver",
5    "state": "CO",
6    "zip": "80231",
7    "country": "USA"
8 }
```

# Where Are We?

JSON Schema Overview

1

Core JSON Schema

2

API Design with JSON Schema

3

# Real World Use Case

**Design/Implement API and Consumer  
in Parallel**

# Our Scenario

**Leverage JSON Schema to  
create a Stub REST API ...  
without any code**

# My JSON Schema Workflow for APIs

**Model JSON Document**

JSONPad

<https://code.google.com/p/json-pad/>

**Generate JSON Schema**

JSON Schema Generator

<http://jsonschema.net/>

**Validate JSON Document**

JSON Validate

<http://jsonvalidate.com/>

JSONLint

<http://jsonlint.com>

**Document JSON Schema (HTML)**

Matic

<https://github.com/mattyod/matic/>

Docson

<https://github.com/lbovet/docson>

# JSONPad Demo

The screenshot shows the JSONPad application interface. The title bar reads "JSONpad". The menu bar includes "File", "Edit", and "Help". The toolbar contains icons for "Tree" (selected), "Format", "Clipboard", and "Tools" (with XML and JSON options). A "Examples" dropdown is also present.

The main area displays a JSON object:

```
2 "about" : "Fred Smith is the CTO of Full Ventures, where he ...",
3 "email" : "fred.smith@fullventures.com",
4 "firstName" : "Fred",
5 "lastName" : "Smith",
6 "tags" : [
7   "JavaScript",
8   "REST",
9   "JSON"
10 ],
11 "company" : "Full Ventures, Inc."
12 }
```

The bottom left pane shows a tree view of the JSON structure under the "JSON" node:

- about
- email
- firstName
- lastName
- tags
- company

The bottom right pane is an "Edit" pane, currently empty.

# JSON Schema Generator Demo

The screenshot shows a web-based JSON Schema Generator tool. At the top, the title "JSON Schema Generator Demo" is displayed. Below it, the browser's address bar shows "jsonschema.net/#/" and the tab title "JSON Schema Generator". The interface includes a toolbar with various icons.

**URL:** `http://jsonsatwork.org`

**JSON:**

```
{
  "about": "Fred Smith is the CTO of Full Ventures, where he ...",
  "email": "fred.smith@fullventures.com",
  "firstName": "Fred",
  "lastName": "Smith",
  "tags": [
    "JavaScript",
    "REST",
    "JSON"
  ],
  "company": "Full Ventures, Inc."
}
```

**Code View** (selected), **Edit View**, **String View**

**Well done! You provided valid JSON.**

**Generate Schema** | **Reset**

**Metadata**

- Include metadata keywords

**General**

- Include default values  
Values are taken from JSON.
- Restrict values to enum  
Uses the default value and null.
- Use absolute IDs
- Force required

**Objects**

- Allow additional properties  
Controls whether it's valid to have additional properties in the object beyond what is defined in the schema.

**Arrays**

- Allow additional items  
Controls whether it's valid to have additional items in the array beyond what is defined in the schema.

**Generated Schema (Code View):**

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "/",
  "type": "object",
  "properties": {
    "about": {
      "id": "about",
      "type": "string"
    },
    "email": {
      "id": "email",
      "type": "string"
    },
    "firstName": {
      "id": "firstName",
      "type": "string"
    },
    "lastName": {
      "id": "lastName",
      "type": "string"
    },
    "tags": {
      "id": "tags",
      "type": "array",
      "items": {
        "id": "2",
        "type": "string"
      },
      "additionalItems": false
    },
    "company": {
      "id": "company",
      "type": "string"
    }
  },
  "additionalProperties": false,
  "required": [
    "about",
    "email",
    "firstName",
    "lastName",
    "tags",
    "company"
  ]
}
```

# JSON Validate Demo

The screenshot shows the JSON Validate demo interface. At the top, the title "JSON Validate" is displayed in a large blue header. Below it, the URL "jsonvalidate.com" is shown in the address bar. The main area is divided into two main sections: "JSON Schema" and "JSON Content".

**JSON Schema:** A code editor containing a JSON schema definition. The schema includes properties like "tags", "company", and "required" fields.

```
22     "tags": "days",
23   "type": "array",
24   "items": {
25     "type": "string"
26   },
27   "additionalItems": false
28 },
29 "company": {
30   "id": "company",
31   "type": "string"
32 }
33 },
34 "additionalProperties": false,
35 "required": [
36   "about",
37   "email",
38   "firstName",
39   "lastName",
40   "tags",
41   "company"
42 ]
43 }
```

**JSON Content:** A code editor containing a JSON object to be validated. The object contains fields such as "about", "email", "firstName", "lastName", "tags", and "company".

```
1 {
2   "about": "Fred Smith is the CTO of Full Ventures, where he ...",
3   "email": "fred.smith@fullventures.com",
4   "firstName": "Fred",
5   "lastName": "Smith",
6   "tags": [
7     "JavaScript",
8     "REST",
9     "JSON"
10 ],
11   "company": "Full Ventures, Inc."
12 }
13
14
15
16
17
18
19
20
21
```

**References:** A sidebar on the left showing numbered references from 1 to 14.

**Results:** A panel on the right displaying the validation status: "Valid".

**Buttons:** At the bottom left are "Validate" and "Reset all" buttons. At the bottom right is a link "Learn more about Using JSON Schema" with a "UJS" logo.

# Matic

```
npm install -g matic
```

```
npm install -g jade
```

```
matic
```

<https://github.com/mattyod/matic>

<https://github.com/mattyod/matic-draft4-example>

# My JSON API Workflow

**Model JSON Document**

JSONPad

[https://  
code.google.com/p/  
json-pad/](https://code.google.com/p/json-pad/)

**Generate JSON Document**

JSON Generator

[http://www.json-  
generator.com/](http://www.json-generator.com/)

**Deploy Stub REST API**

JSON Server

[https://github.com/  
typicode/json-server](https://github.com/typicode/json-server)

# JSON Generator Demo

The screenshot shows a web application titled "JSON GENERATOR" on the "www.json-generator.com" website. The interface includes a toolbar with "Generate", "Reset", "Try out beta!", "Help", and "Feedback" buttons. The main area contains a code editor with the following JSON template:

```
1 // Template for http://www.json-generator.com/
2
3 [ 
4   {{repeat(3)}}, {
5     id: '{{integer()}}',
6     picture: 'http://placehold.it/32x32',
7     name: '{{firstName()}}',
8     lastName: '{{surname()}}',
9     company: '{{company()}}',
10    email: '{{email()}}',
11    about: '{{lorem(1, "paragraphs")}}'
12  }
13 ]
```

To the right of the code editor is a preview area with a placeholder message: "Click 'Generate' and wait for the magic to happen". At the bottom, there are social sharing links for Twitter, Facebook, and LinkedIn.

Created by [Vazha Omanashvili](#), Sponsored by [Runscope API Tools](#)

Follow Tweet 1,676 Like Share 1k

# JSON Server

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

```
json-server speakers.json
```

<http://localhost:3000/speakers>

<https://github.com/typicode/json-server>

# Our Agenda

**JSON Schema Overview**

1

**Core JSON Schema**

2

**API Design with JSON Schema**

3

# Questions?

**Tom Marrs**

**@TomMarrs**

**thomasamarrs@comcast.net**



# JSON Resources

JSON Spec - <http://tools.ietf.org/html/rfc7159>

ECMA 404 - <http://www.ecma-international.org/publications/standards/Ecma-404.htm>

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

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

# JSON Resources

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

JSONPad - <https://code.google.com/p/json-pad/>

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

# JSON Schema Resources

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

JSON Schema Spec - <http://json-schema.org/latest/json-schema-core.html>

JSON Schema Validation Spec - <http://json-schema.org/latest/json-schema-validation.html>

JSON Hyper-Schema Spec - <http://json-schema.org/latest/json-schema-hypermedia.html>

# JSON Schema Resources

Using JSON Schema - <http://usingjsonschema.com/>

JSON Validate - <http://jsonvalidate.com/>

Understanding JSON Schema - <http://spacetelescope.github.io/understanding-json-schema/>

jsonschema.net - <http://jsonschema.net/>

# JSON Schema Resources

**JSON Schema GitHub Repo - <https://github.com/kriszyp/json-schema>**

**JSON Schema Google Group - <https://groups.google.com/forum/#!forum/json-schema>**

**Put Some JSON Schema in your life - <https://kreuzwerker.de/en/blog/posts/put-some-json-schema-in-your-life>**

**JSON Pointer Spec - <https://tools.ietf.org/html/rfc6901>**