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

Where Are We?

JSON Schema Overview 1

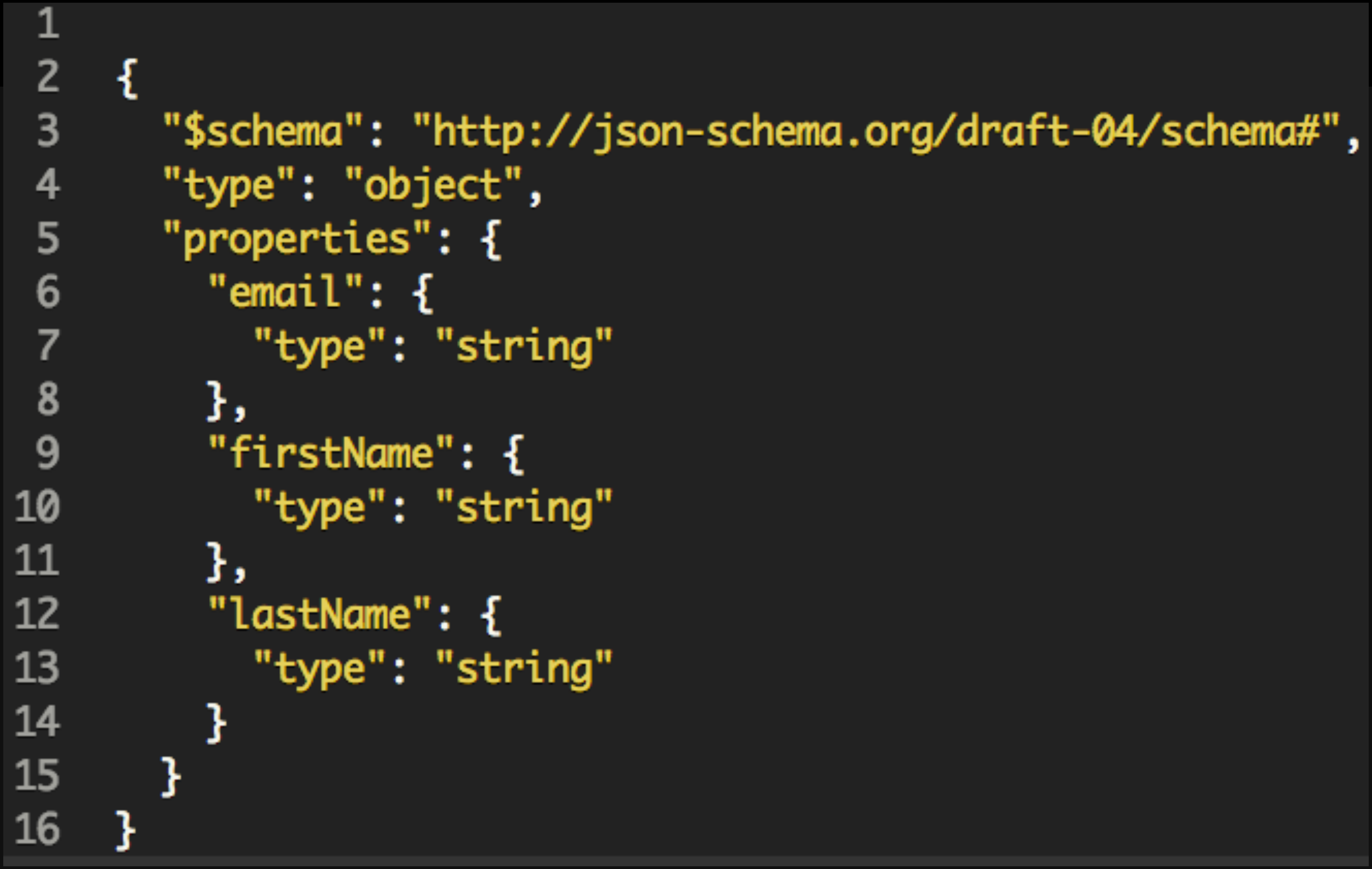
Core JSON Schema 2

API Design with JSON Schema 3

What is JSON Schema?

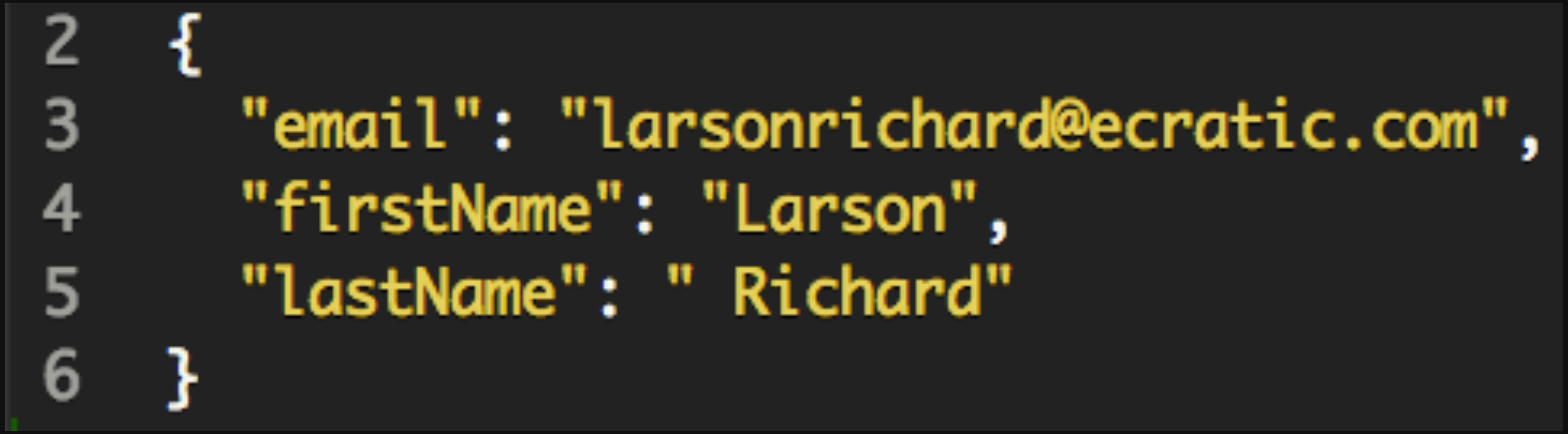
Validate Structure + Format

Basic JSON Schema



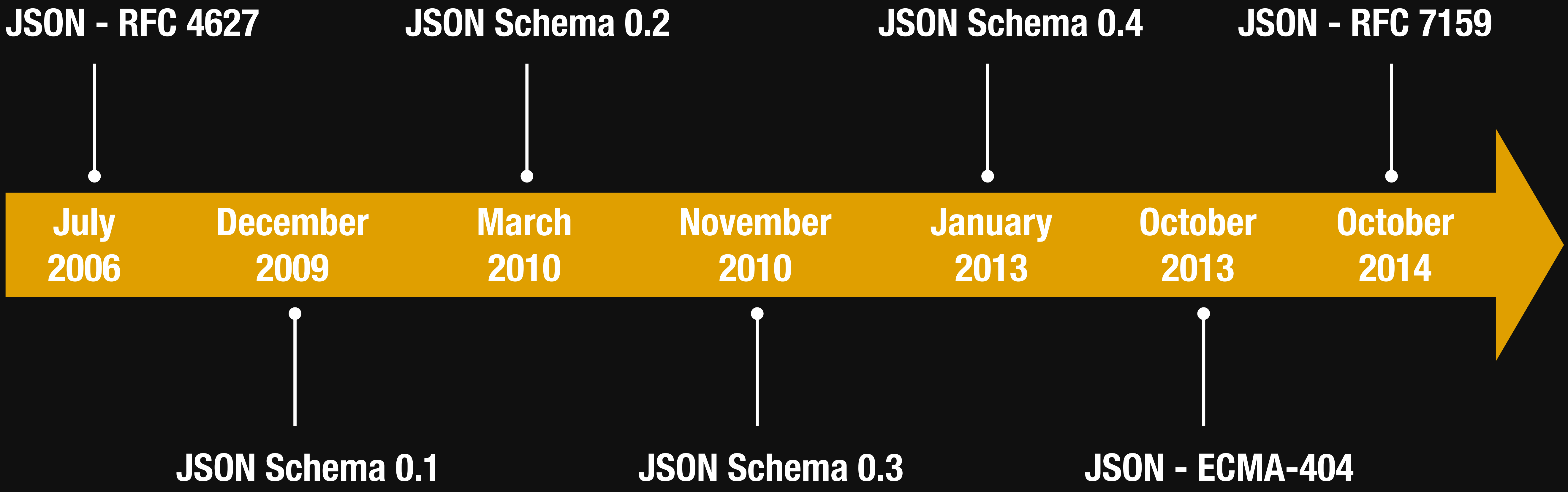
```json

{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "email": {
      "type": "string"
    },
    "firstName": {
      "type": "string"
    },
    "lastName": {
      "type": "string"
    }
  }
}
```

Basic JSON - Document



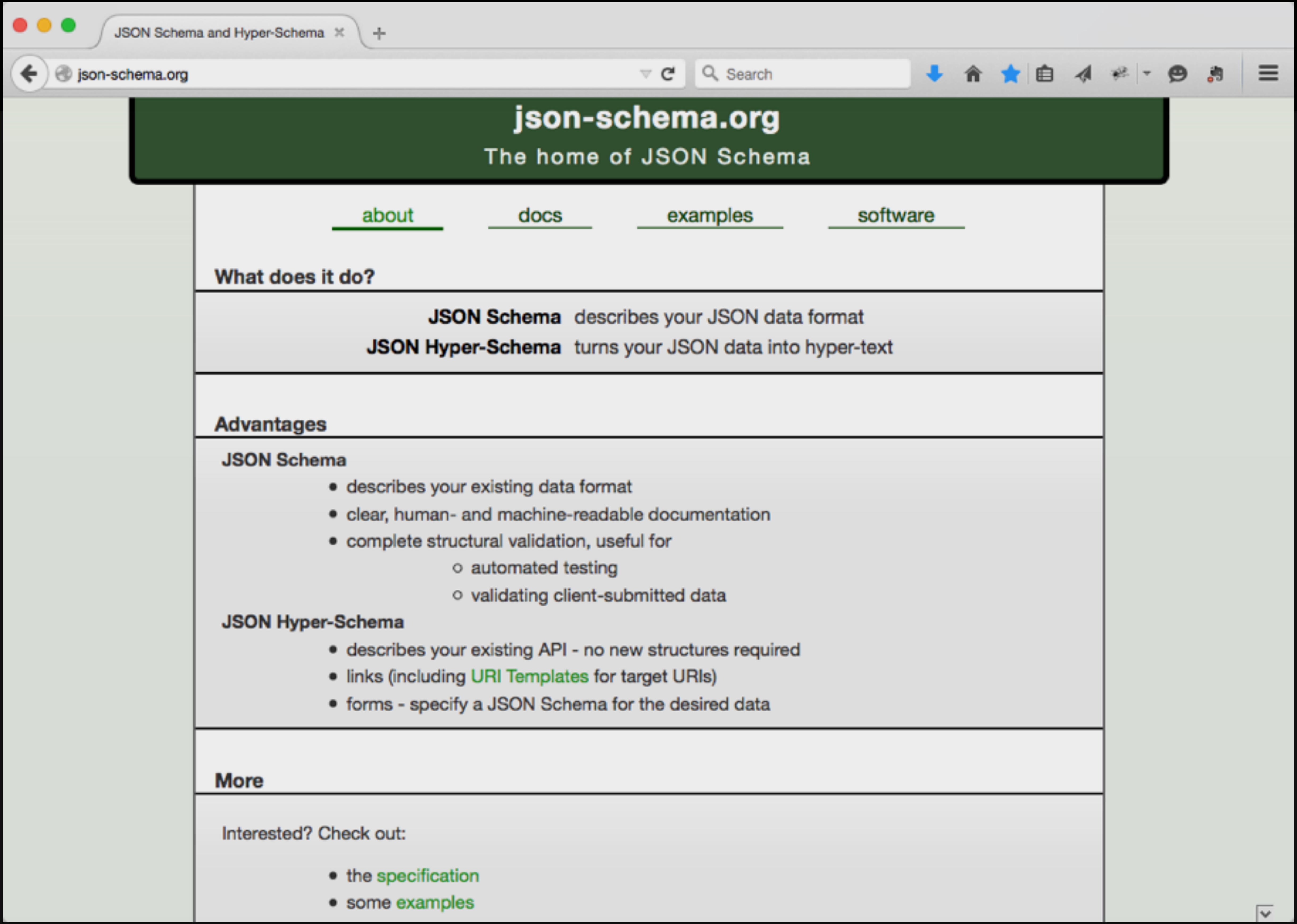
```
{
  "email": "larsonrichard@ecratic.com",
  "firstName": "Larson",
  "lastName": " Richard"
}
```

JSON Schema Timeline - When?



The Journey …

json-schema.org



json-schema.org

The home of JSON Schema

about docs examples software

What does it do?

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

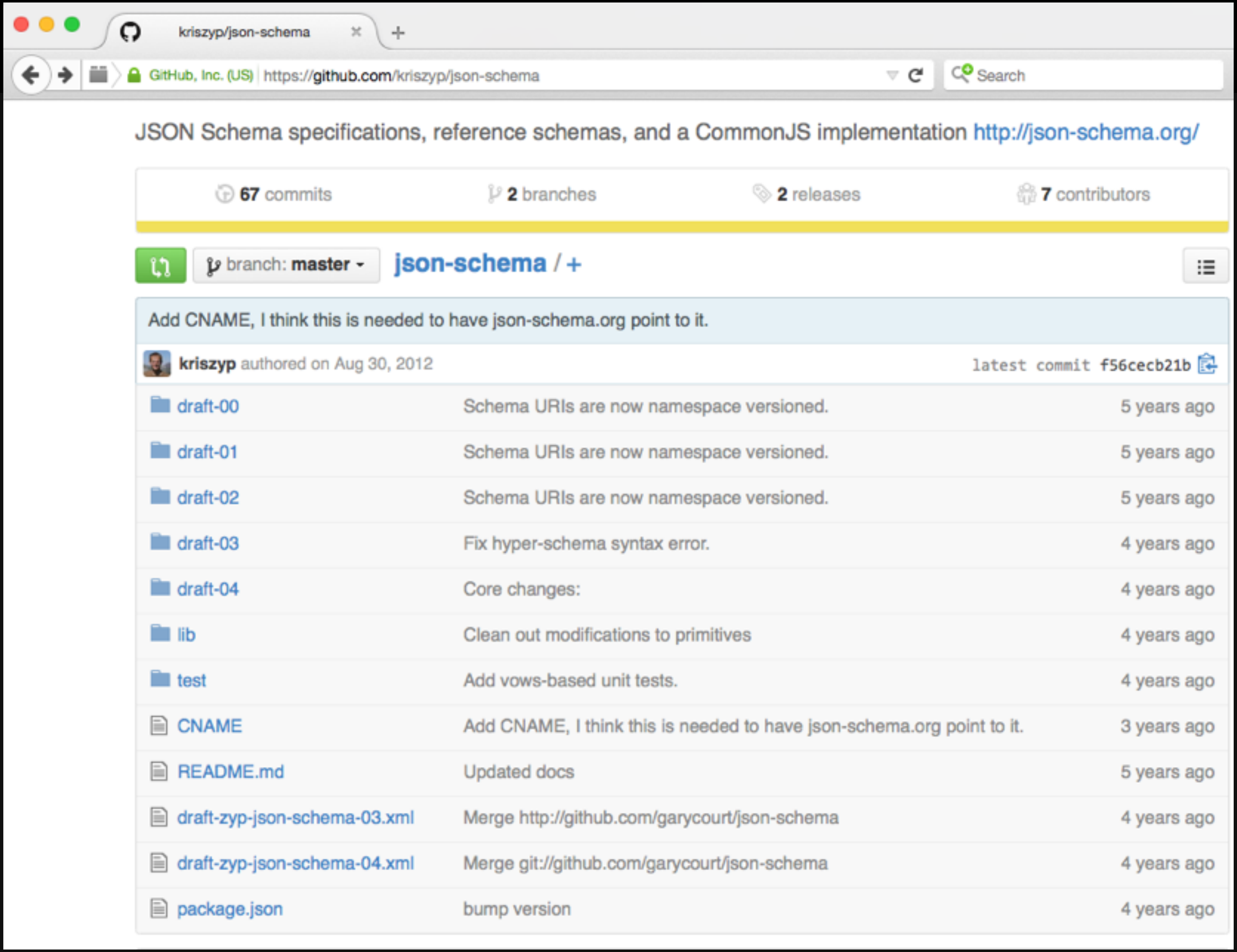
- describes your existing API - no new structures required
- links (including URI Templates for target URIs)
- forms - specify a JSON Schema for the desired data

More

Interested? Check out:

- the specification
- some examples

JSON Schema on GitHub



JSON Schema specifications, reference schemas, and a CommonJS implementation http://json-schema.org/

67 commits | 2 branches | 2 releases | 7 contributors

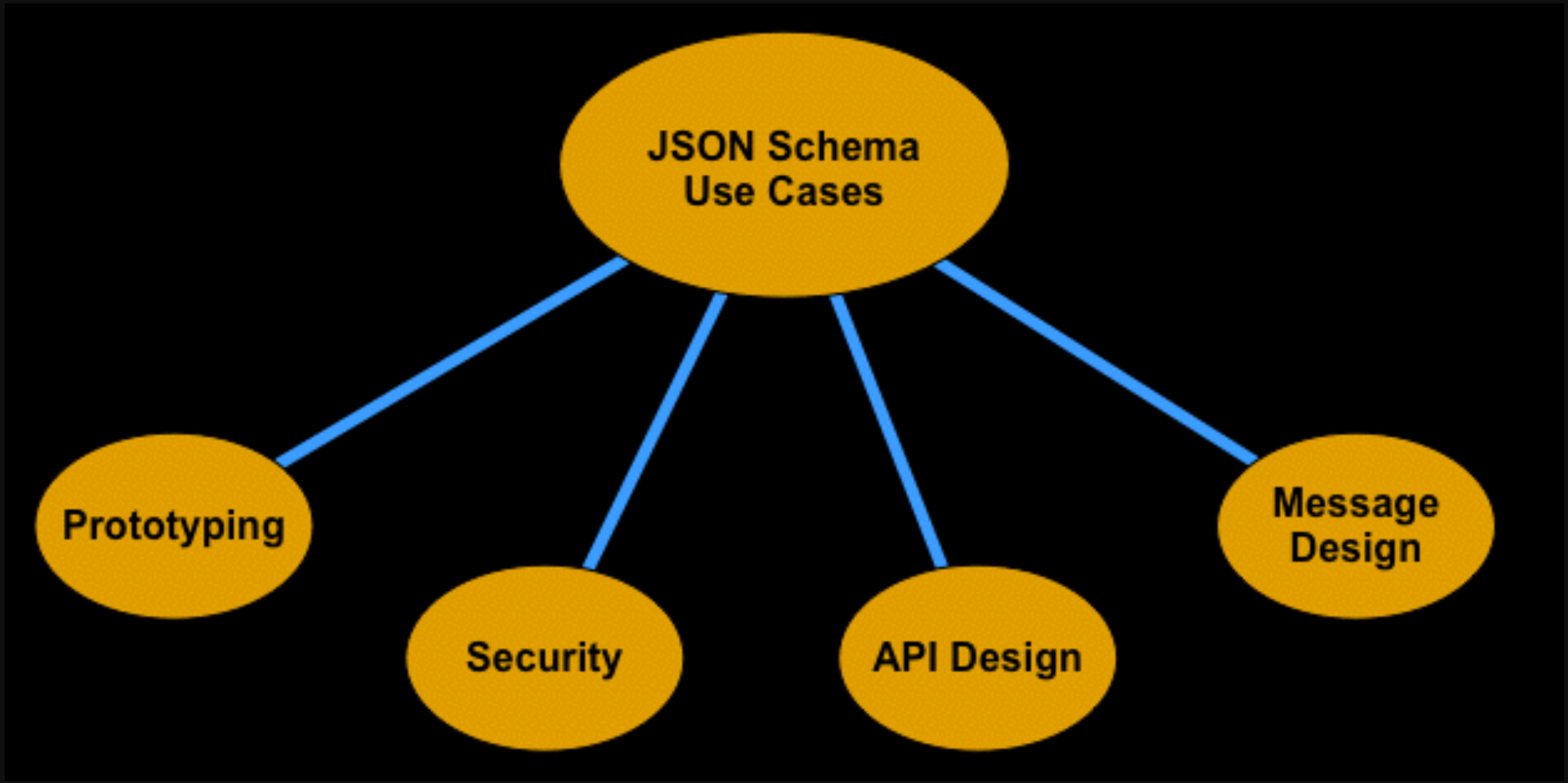
branch: master | json-schema / +

Add CNAME, I think this is needed to have json-schema.org point to it.

kriszyp authored on Aug 30, 2012 — latest commit f56cecb21b

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?

MuleSoft™

heroku

node JS

Swagger

IBM.
developerWorks®

Google
Developers

Why isn’t JSON Validation Enough?

Semantics	Structure
Schema + Instance Document	Instance Document
Meaning	Well-formed - Valid JSON
Ex: Person, Order	Syntax

Haven't We Seen This Before?

JSON Schema	XML Schema
No Reference	Instance Document references Schema
No Namespace - Yes!	Namespace Misery
.json	.xsd

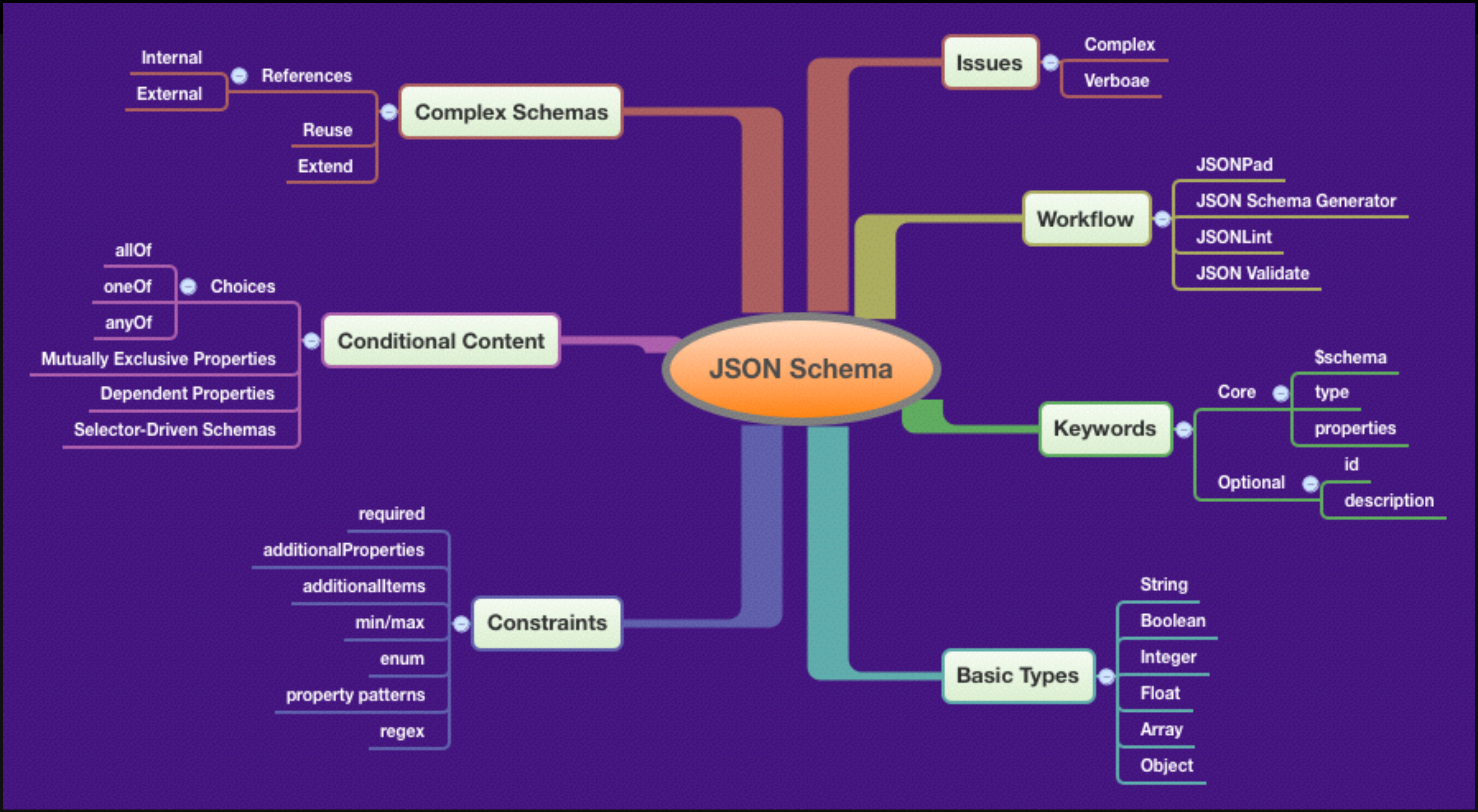
Where Are We?

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/

Generate JSON Document

JSON Generator

http://jsonschema.net/

Validate JSON Document

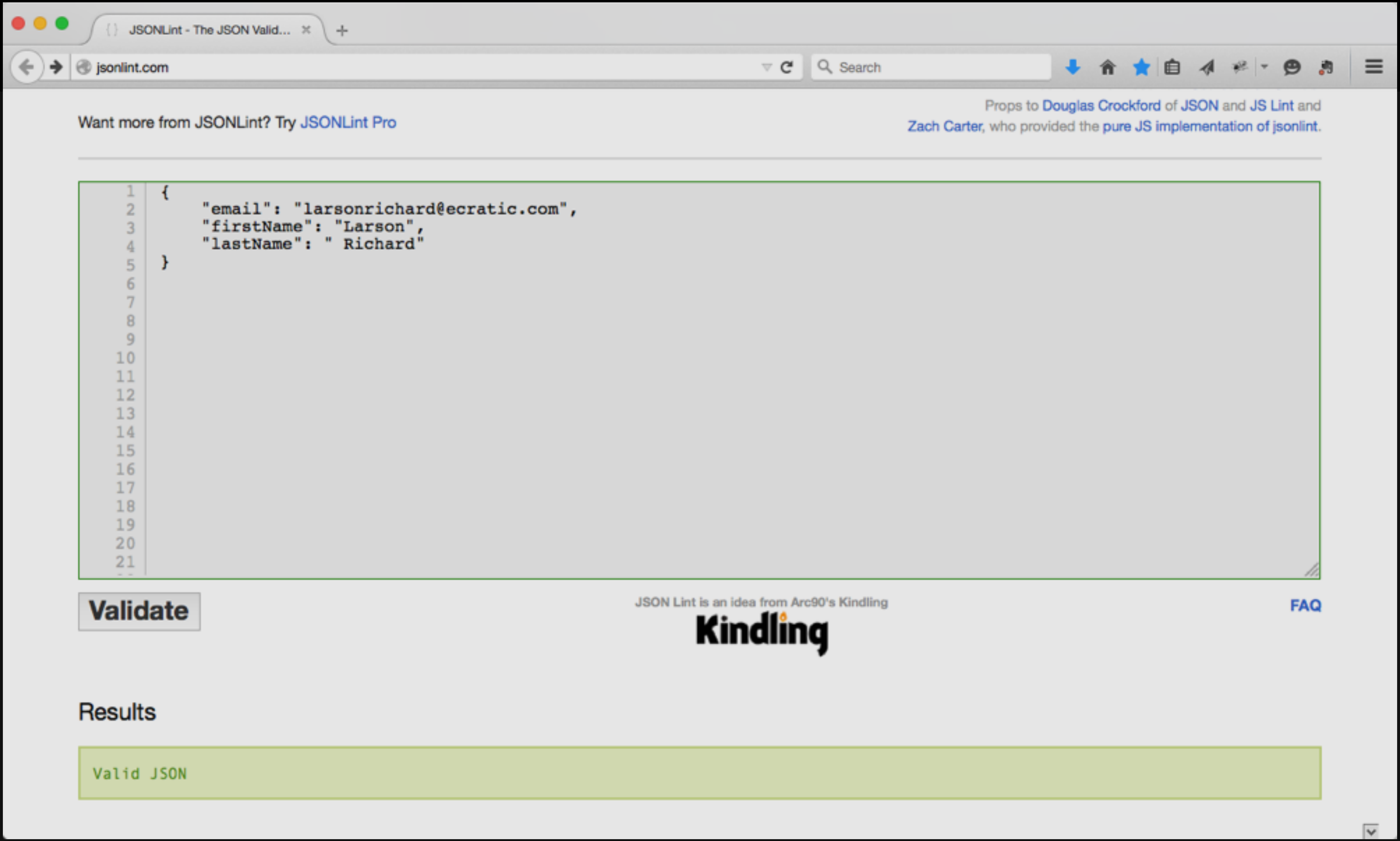
JSON Validate

http://jsonvalidate.com/

JSONLint

http://jsonlint.com

JSONLint


JSONLint - The JSON Valid...
jsonlint.com
Search

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.

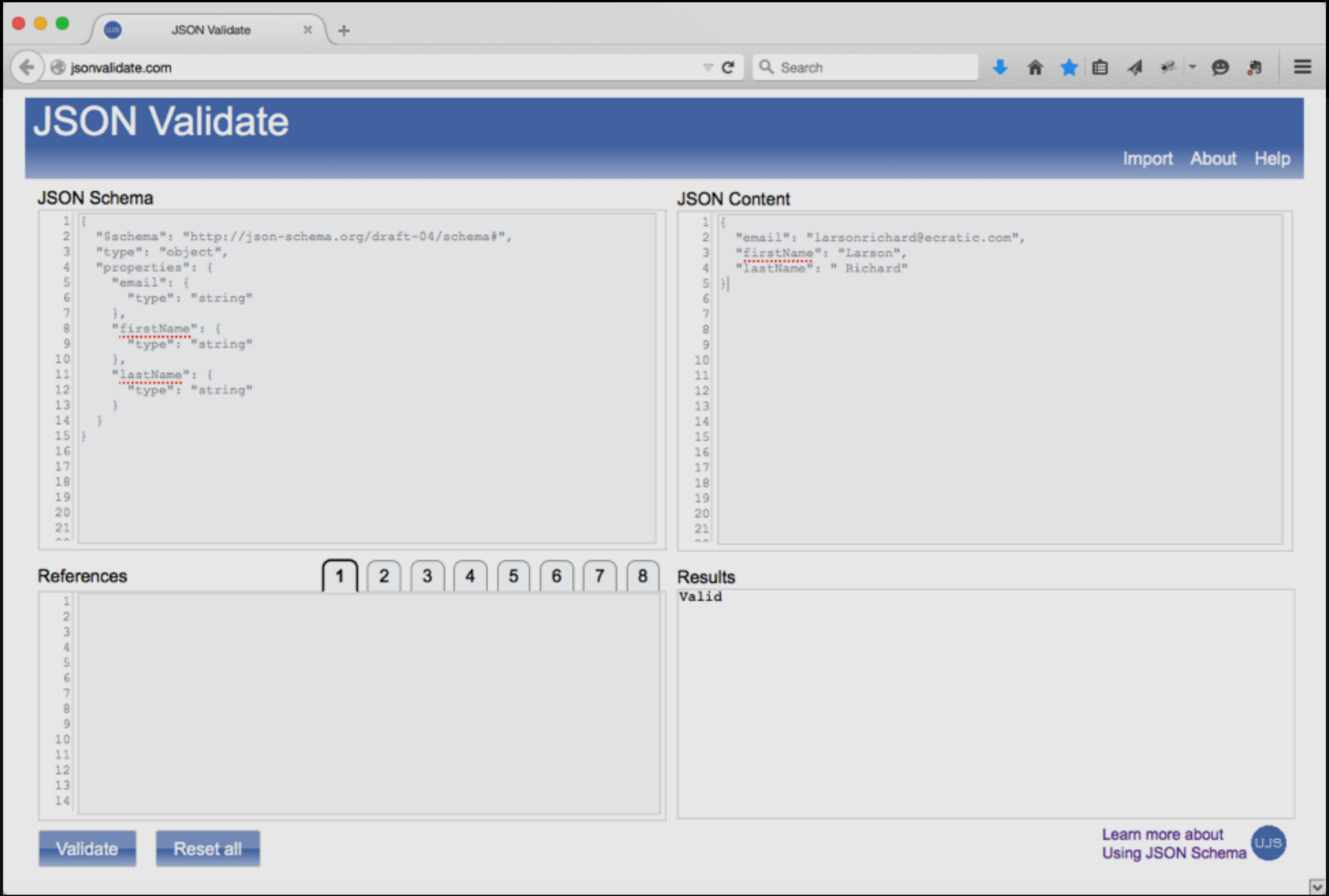
```
{
    "email": "larsonrichard@ecratic.com",
    "firstName": "Larson",
    "lastName": " Richard"
}
```

Validate
JSON Lint is an idea from Arc90's Kindling
Kindling
FAQ

Results

```
Valid JSON
```

JSON Validate



```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "email": {
      "type": "string"
    },
    "firstName": {
      "type": "string"
    },
    "lastName": {
      "type": "string"
    }
  }
}
```

```
{
  "email": "larsonrichard@ecratic.com",
  "firstName": "Larson",
  "lastName": " Richard"
}
```

```
Valid
```

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

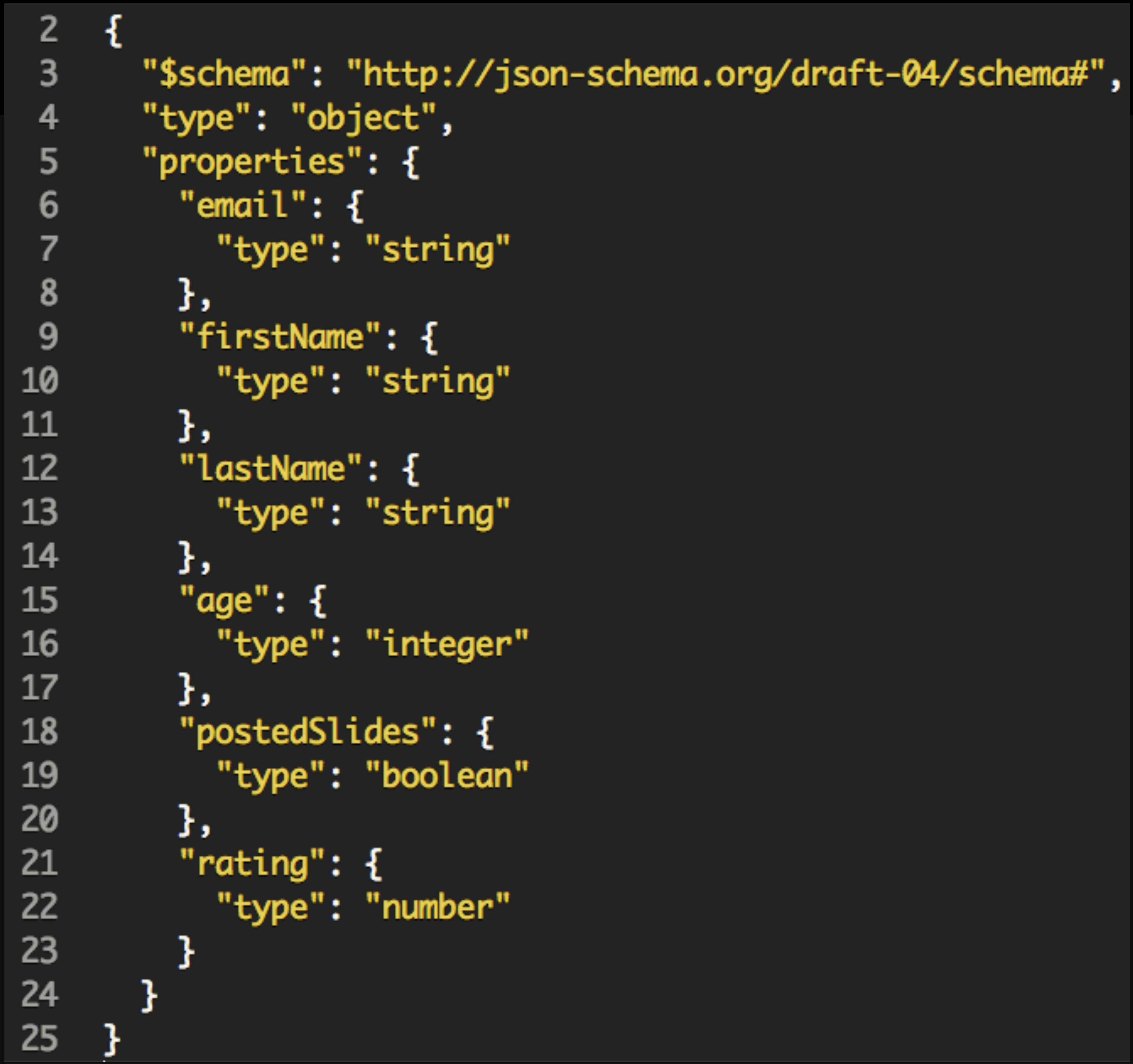
Basic Keywords

Keyword	Definition
`$schema`	Specify JSON Schema version - `“$schema": "http://json-schema.org/draft-04/schema#"`
`type`	The data type - `“type”: “string”`
`properties`	The fields for an object

Optional Keywords

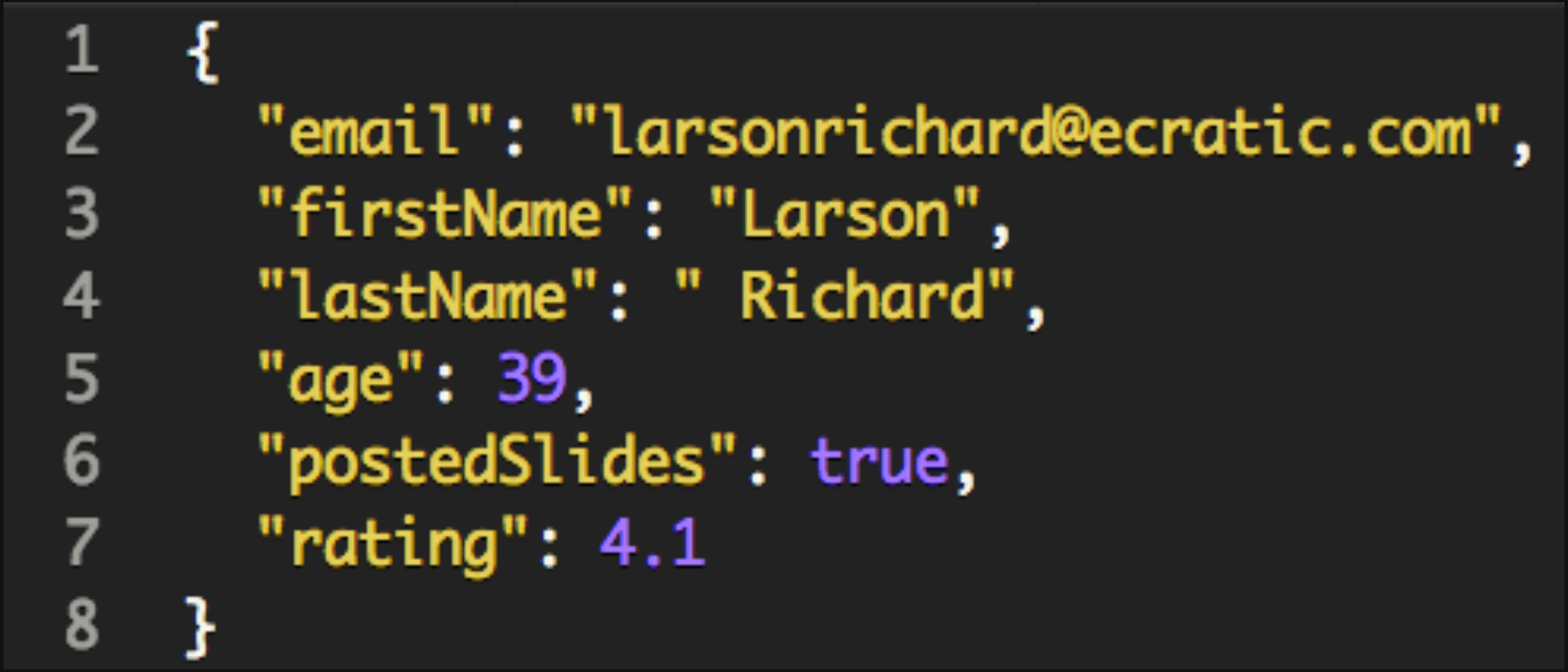
Keyword	Definition
`id`	(1): Path to field (2): URI to Schema
`description`	For documentation

Basic Types - JSON Schema



```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "email": {
      "type": "string"
    },
    "firstName": {
      "type": "string"
    },
    "lastName": {
      "type": "string"
    },
    "age": {
      "type": "integer"
    },
    "postedSlides": {
      "type": "boolean"
    },
    "rating": {
      "type": "number"
    }
  }
}
```

Basic Types - JSON Document

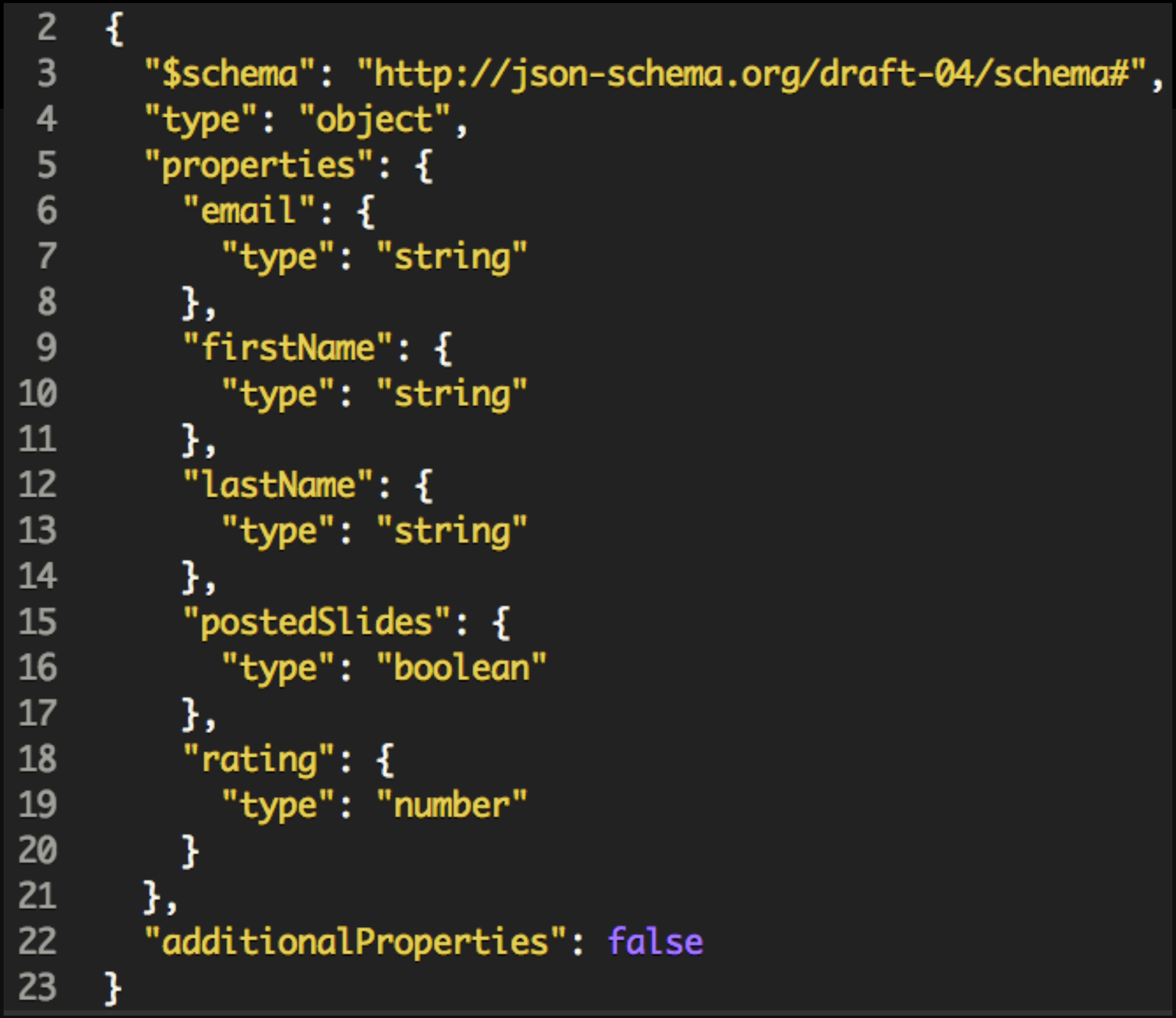


```json
{
  "email": "larsonrichard@ecratic.com",
  "firstName": "Larson",
  "lastName": " Richard",
  "age": 39,
  "postedSlides": true,
  "rating": 4.1
}
```

Where’s the Validation?

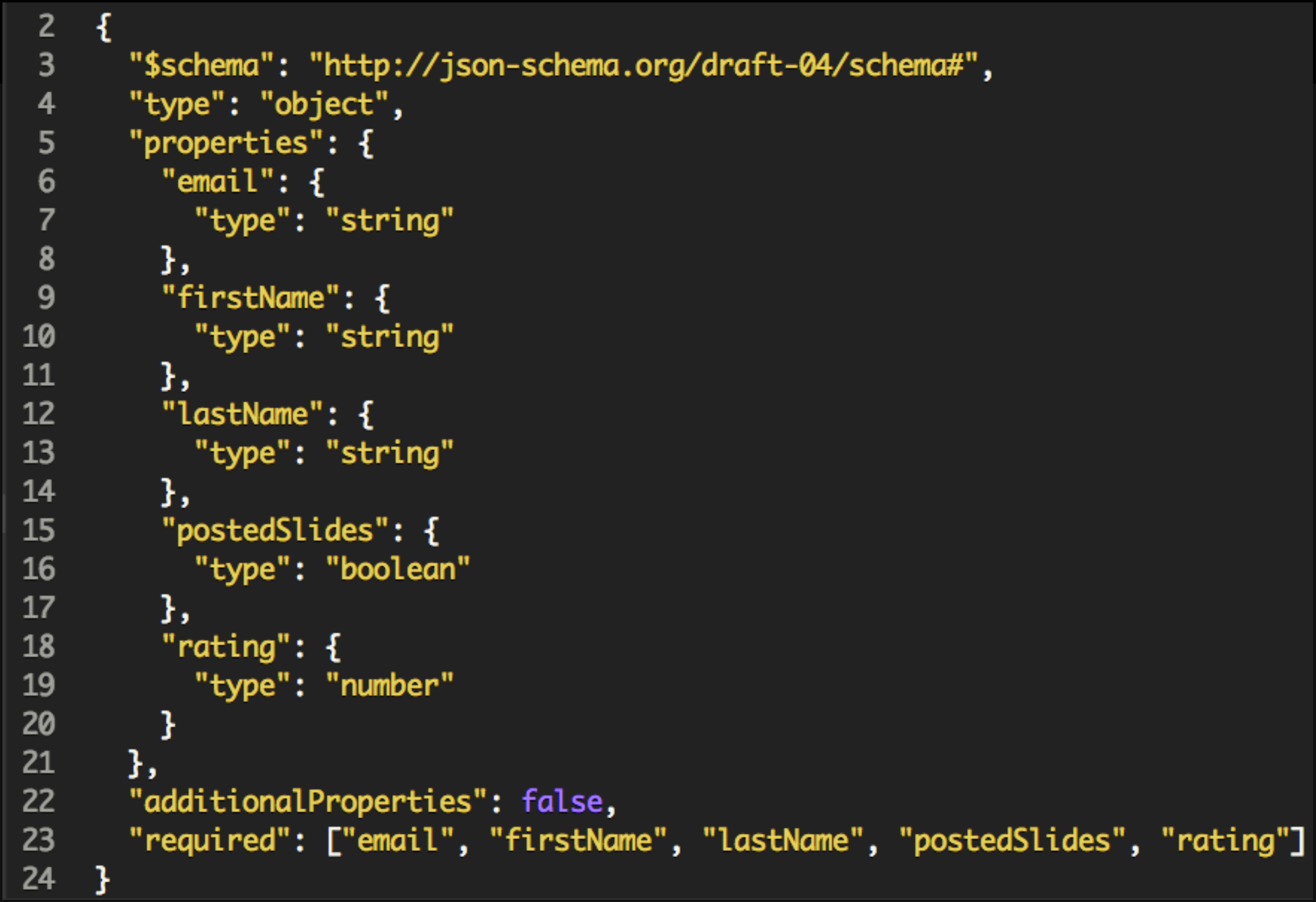
Keyword	Definition
`additionalProperties`	enable/disable additional fields in an object
`required`	Which fields are required
`additionalItems`	enable/disable additional array elements

Basic Types Validation - JSON Schema



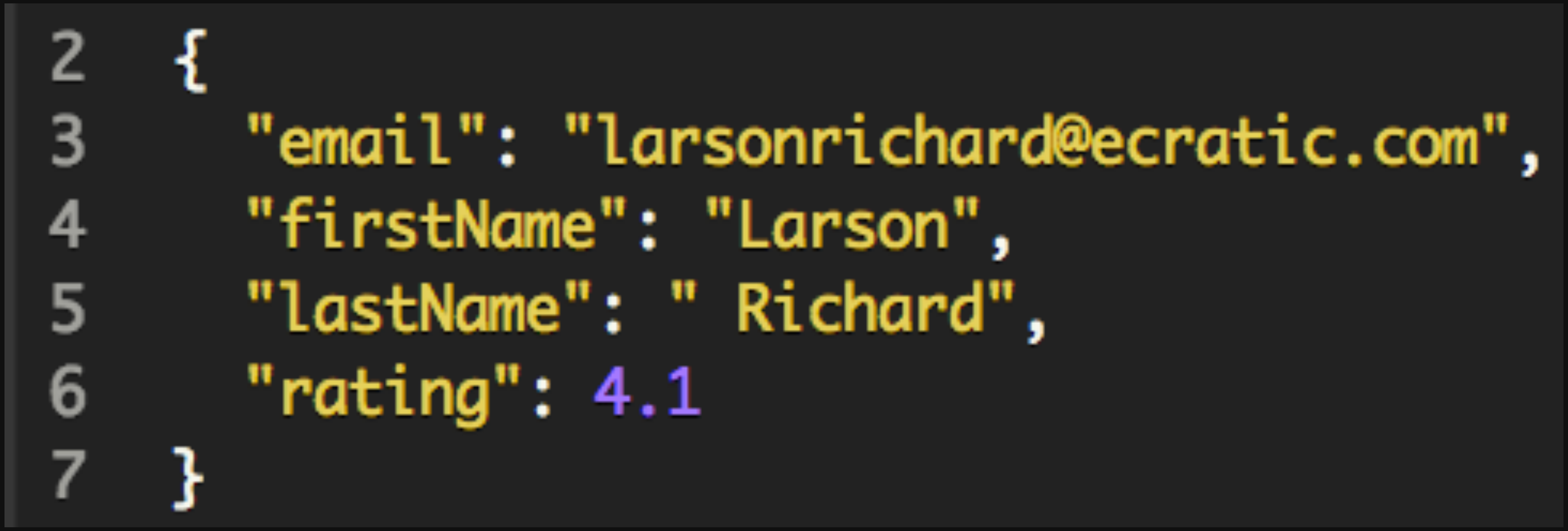
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "email": {
      "type": "string"
    },
    "firstName": {
      "type": "string"
    },
    "lastName": {
      "type": "string"
    },
    "postedSlides": {
      "type": "boolean"
    },
    "rating": {
      "type": "number"
    }
  },
  "additionalProperties": false
}
```

Validation with Required - JSON Schema



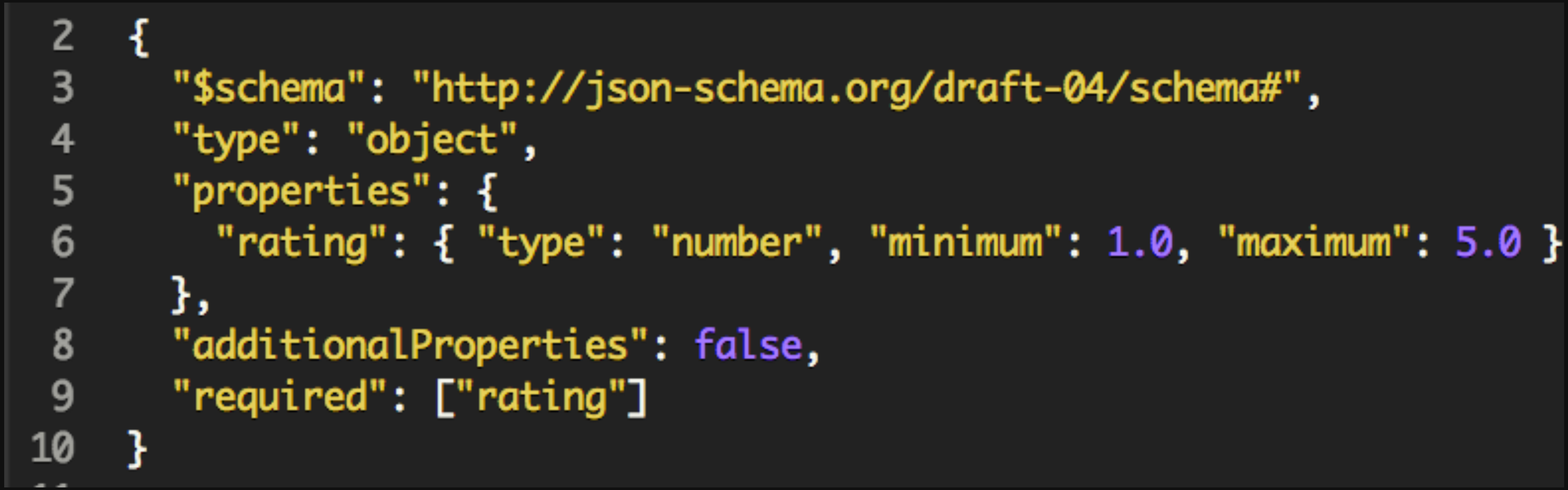
```json
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "email": {
      "type": "string"
    },
    "firstName": {
      "type": "string"
    },
    "lastName": {
      "type": "string"
    },
    "postedSlides": {
      "type": "boolean"
    },
    "rating": {
      "type": "number"
    }
  },
  "additionalProperties": false,
  "required": ["email", "firstName", "lastName", "postedSlides", "rating"]
}
```

Validation with Required - JSON Doc



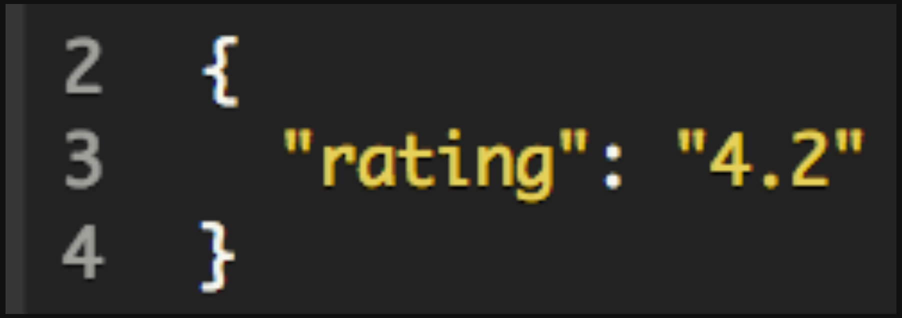
```
{
  "email": "larsonrichard@ecratic.com",
  "firstName": "Larson",
  "lastName": " Richard",
  "rating": 4.1
}
```

Number min/max - JSON Schema



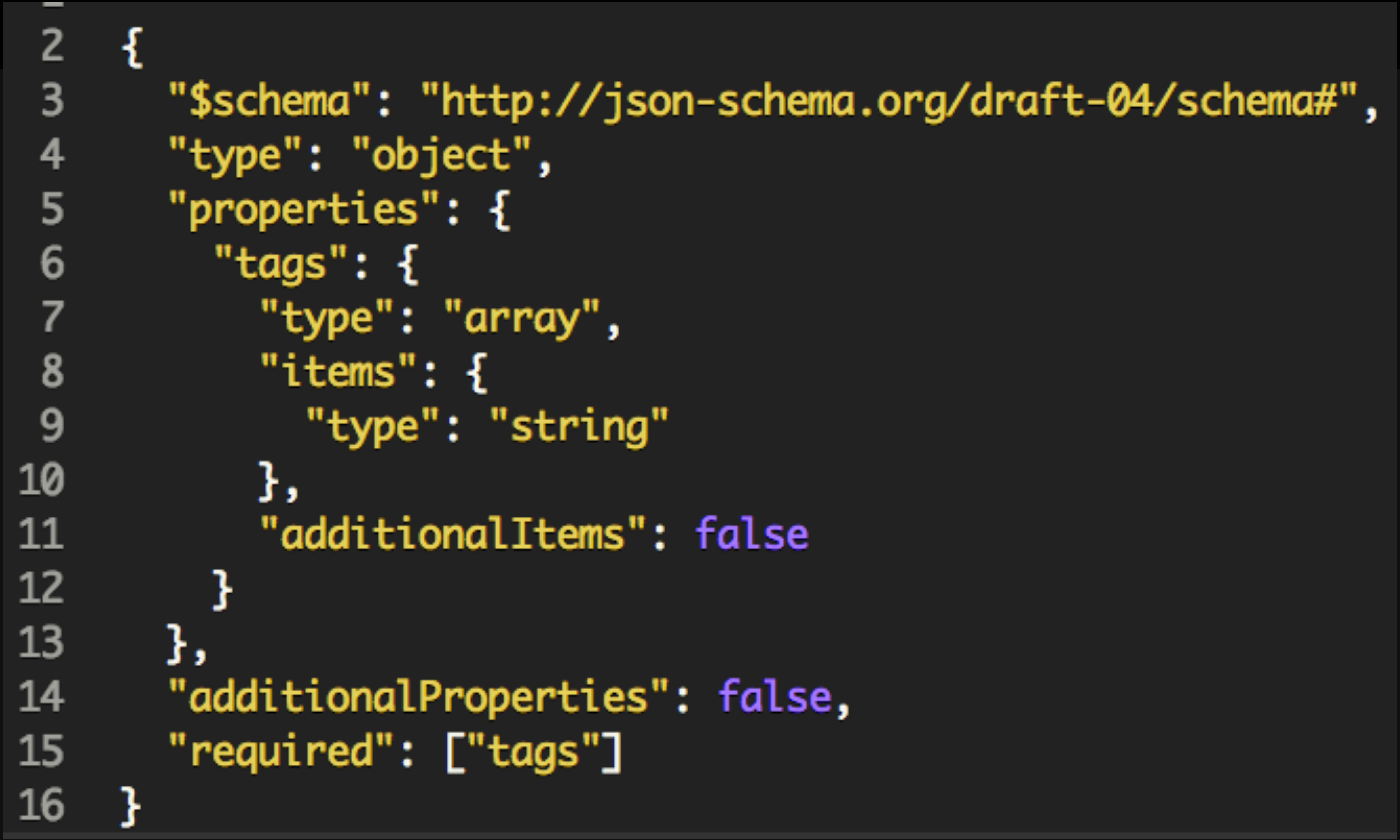
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "rating": { "type": "number", "minimum": 1.0, "maximum": 5.0 }
  },
  "additionalProperties": false,
  "required": ["rating"]
}
```

Number min/max - JSON Doc



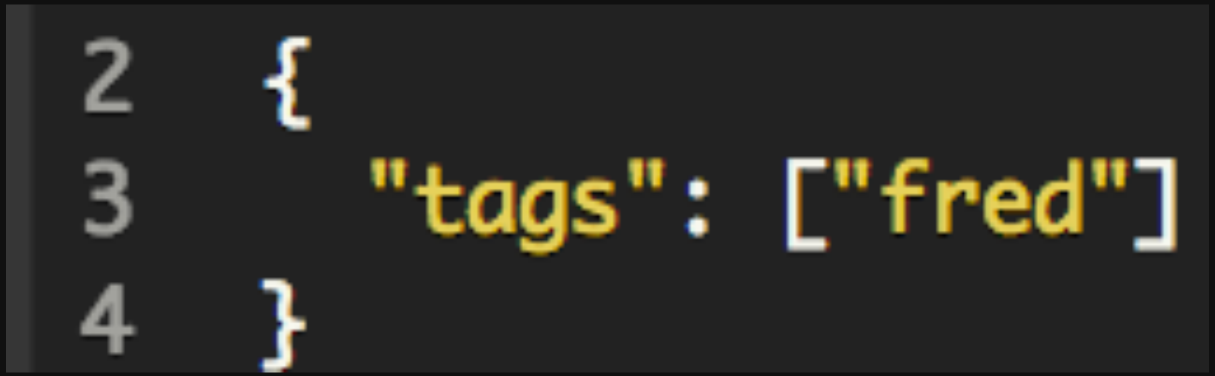
```
{
  "rating": "4.2"
}
```

Simple Array - JSON Schema



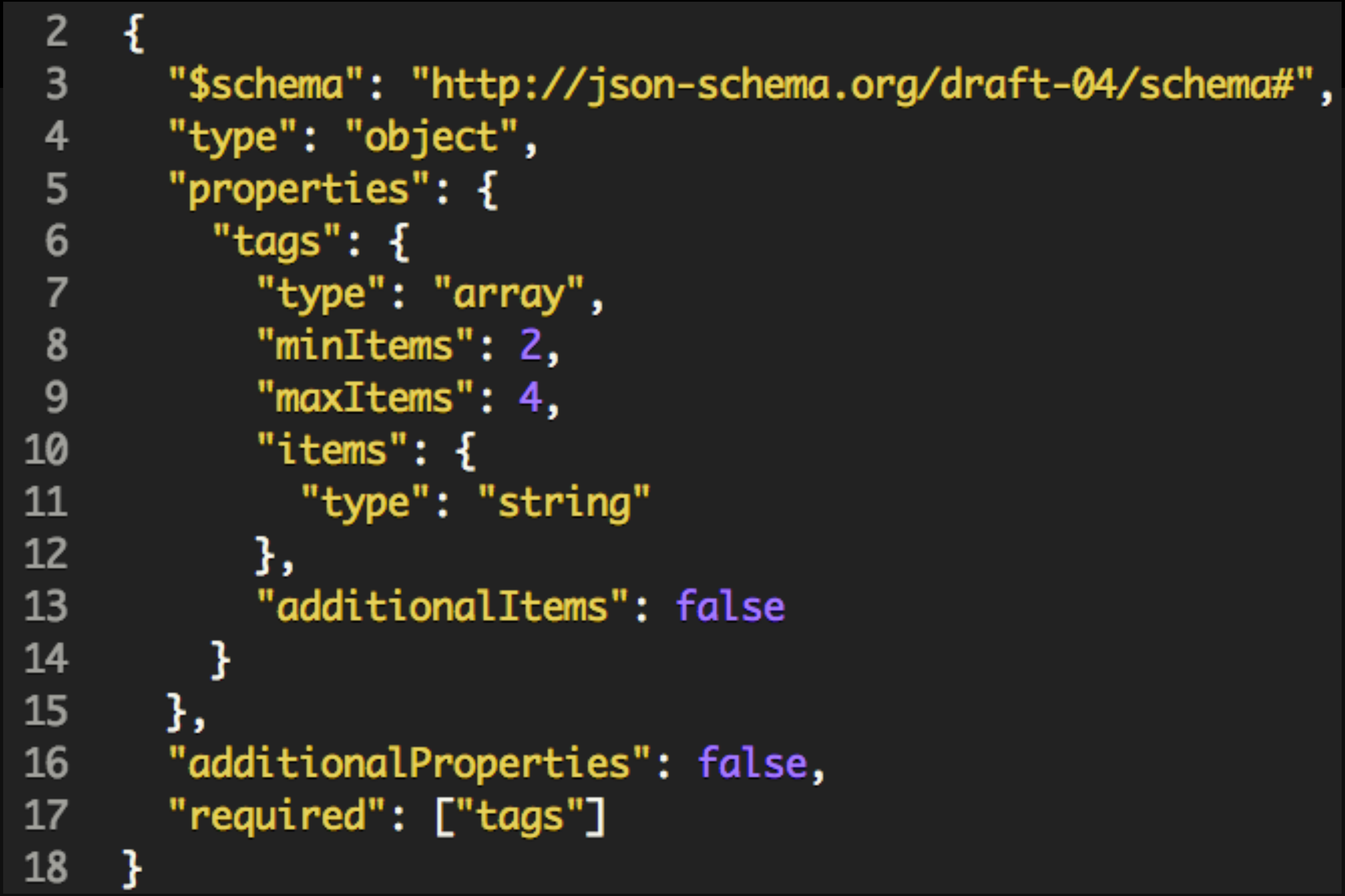
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "tags": {
      "type": "array",
      "items": {
        "type": "string"
      },
      "additionalItems": false
    }
  },
  "additionalProperties": false,
  "required": ["tags"]
}
```

Simple Array - JSON Doc



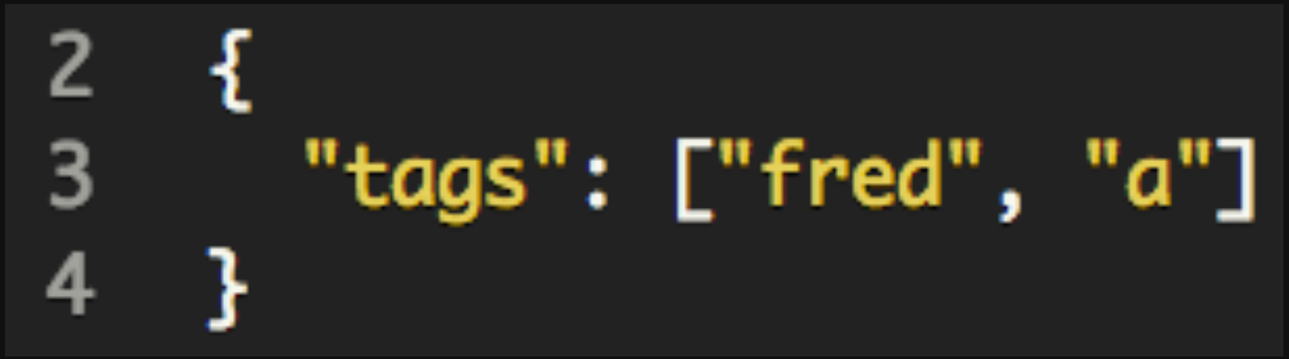
```
{
  "tags": ["fred"]
}
```

Array min/max - JSON Schema



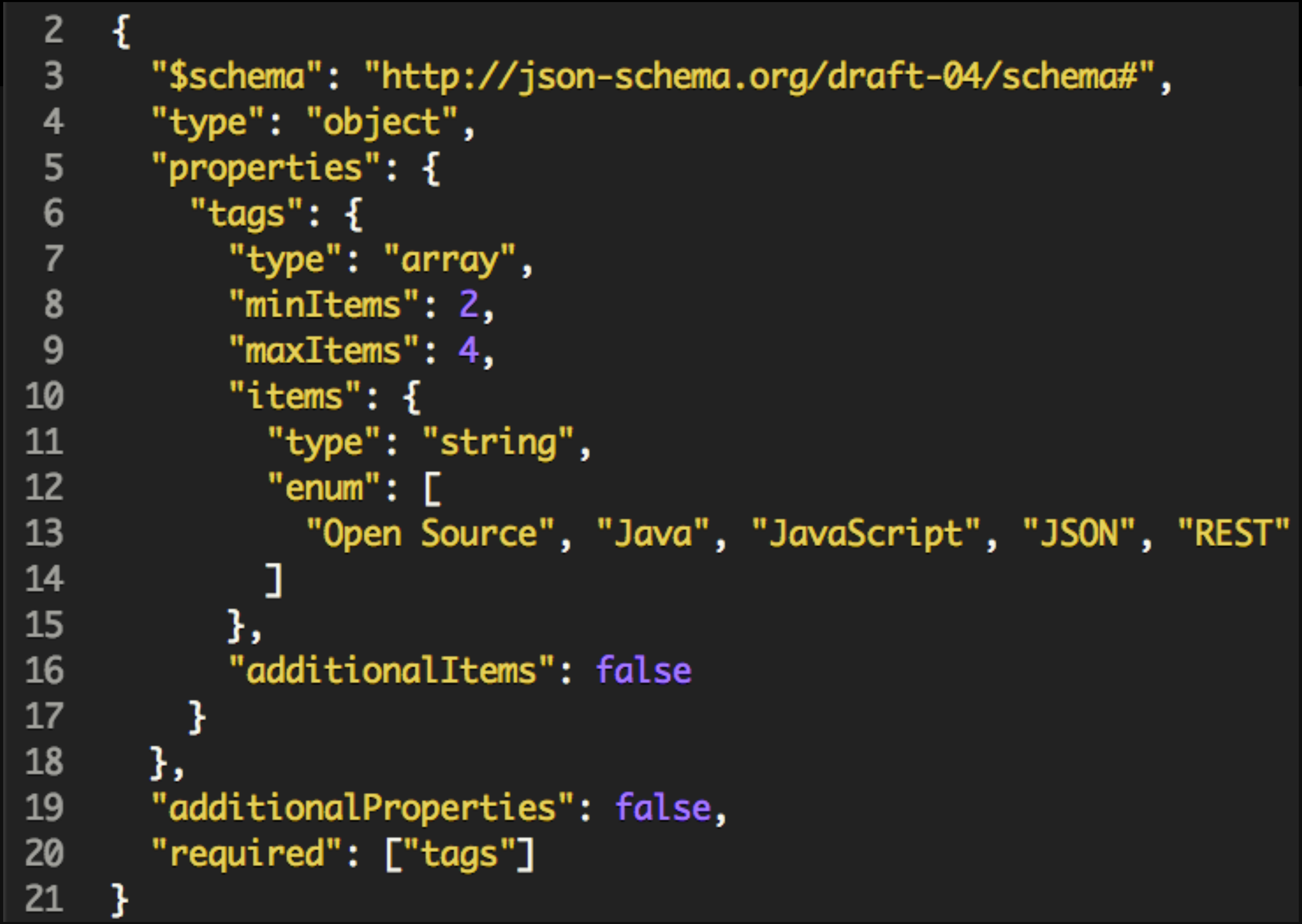
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "tags": {
      "type": "array",
      "minItems": 2,
      "maxItems": 4,
      "items": {
        "type": "string"
      },
      "additionalItems": false
    }
  },
  "additionalProperties": false,
  "required": ["tags"]
}
```

Array min/max - JSON Doc



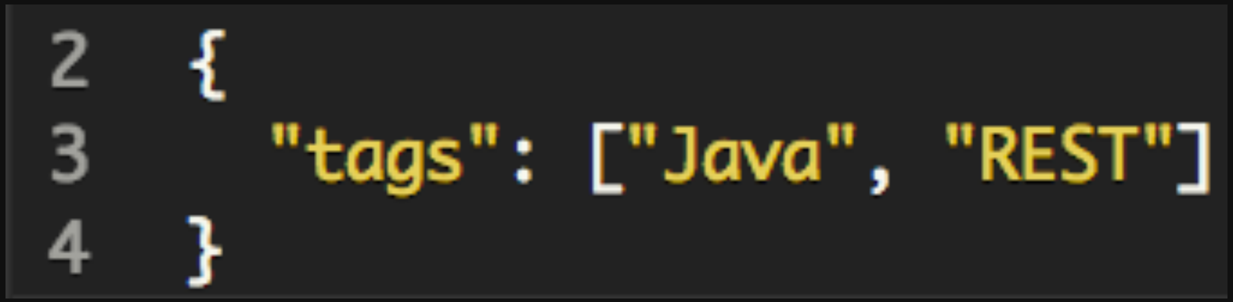
```
{
  "tags": ["fred", "a"]
}
```

Array Enum - JSON Schema



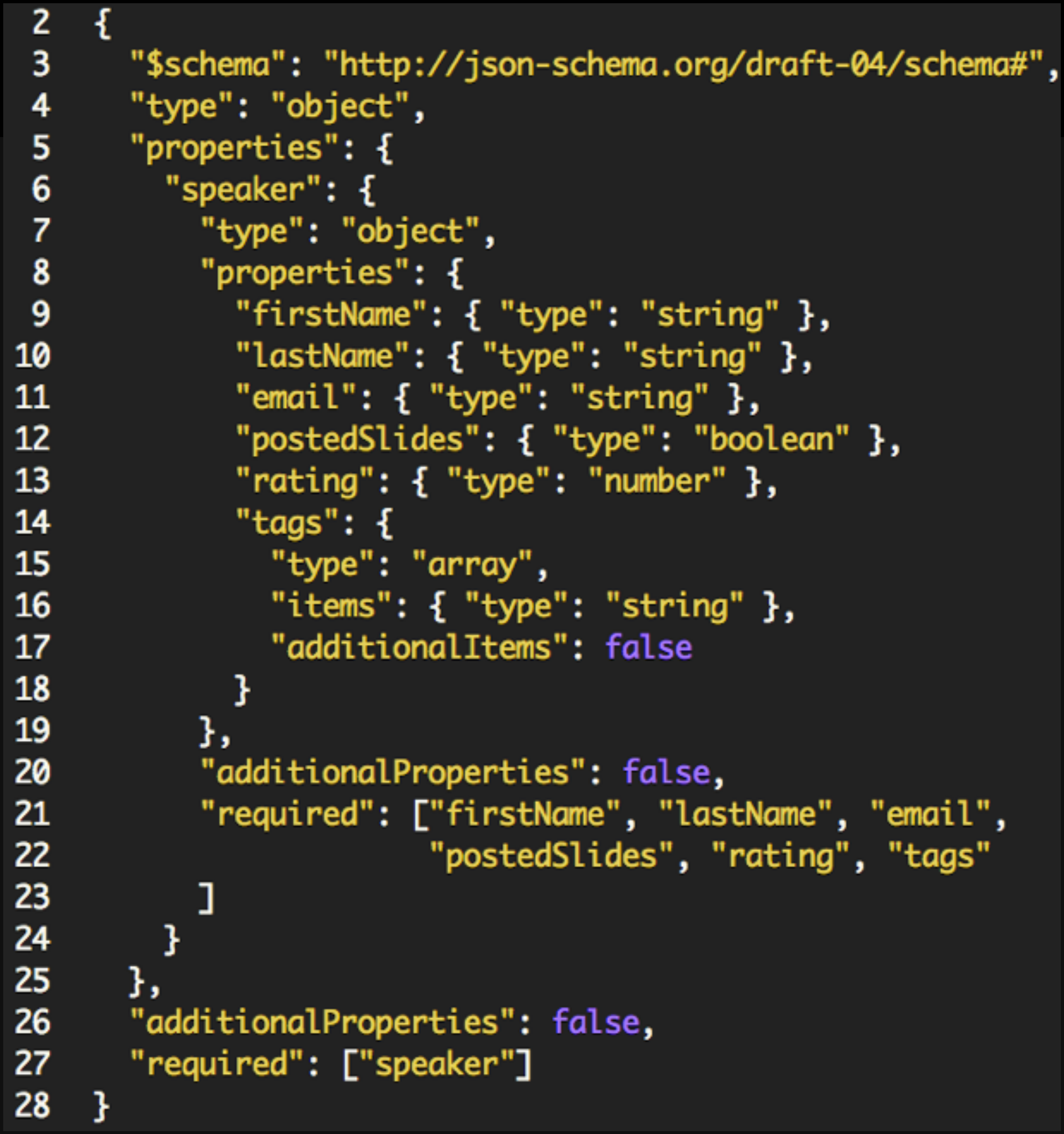
```json
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "tags": {
      "type": "array",
      "minItems": 2,
      "maxItems": 4,
      "items": {
        "type": "string",
        "enum": [
          "Open Source", "Java", "JavaScript", "JSON", "REST"
        ]
      },
      "additionalItems": false
    }
  },
  "additionalProperties": false,
  "required": ["tags"]
}
```

Array Enum - JSON Doc



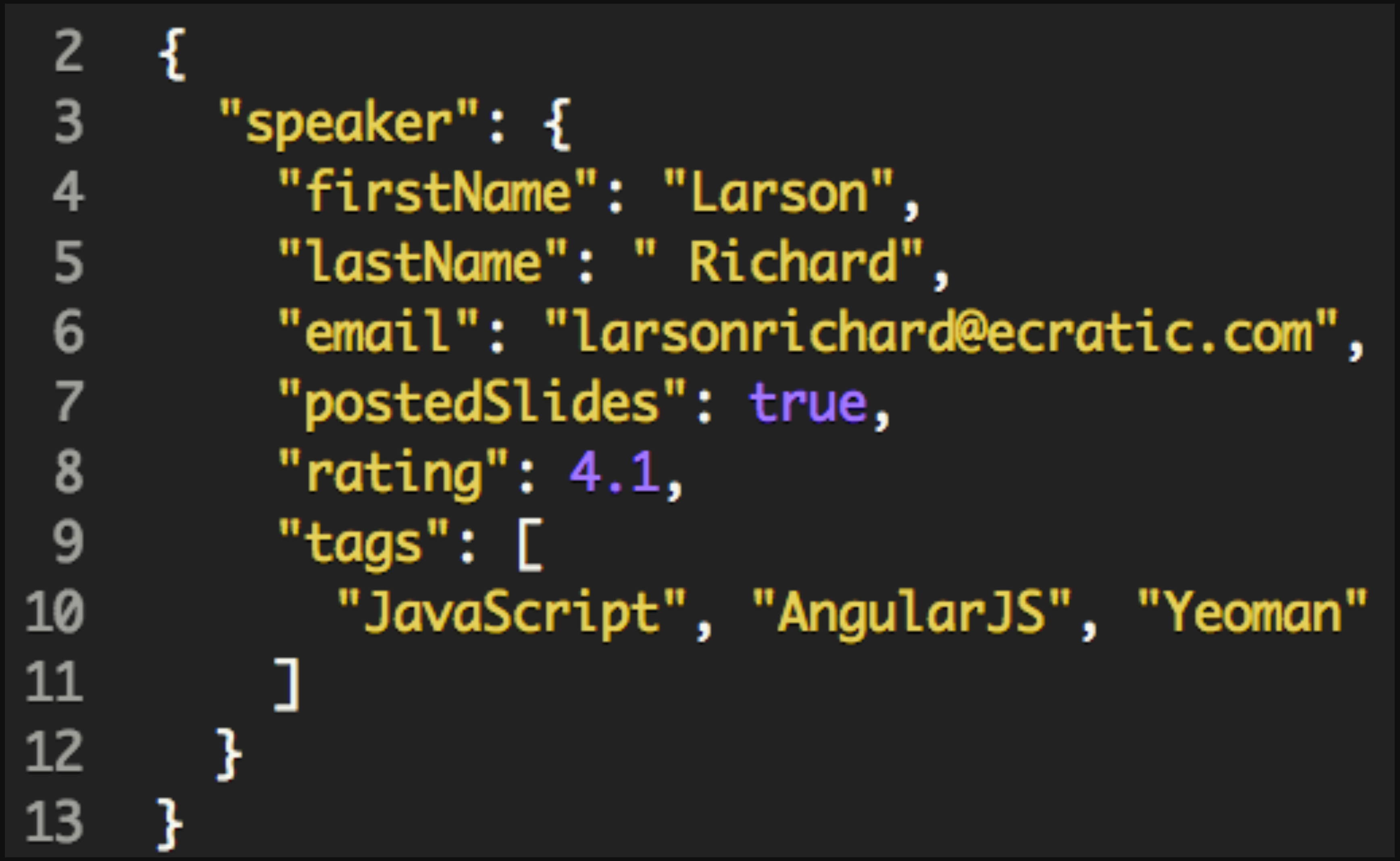
```
{
  "tags": ["Java", "REST"]
}
```

Named Object - JSON Schema



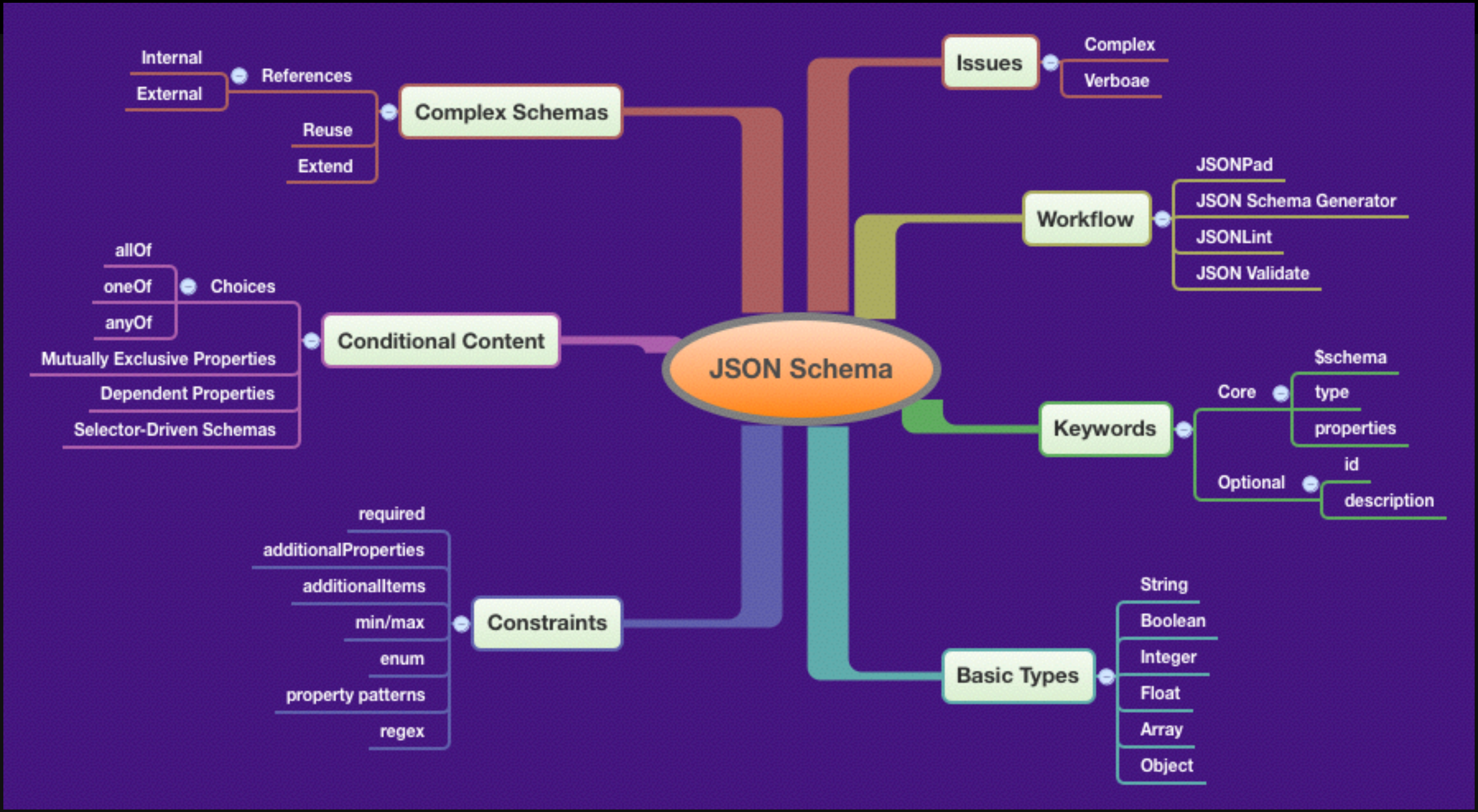
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "speaker": {
      "type": "object",
      "properties": {
        "firstName": { "type": "string" },
        "lastName": { "type": "string" },
        "email": { "type": "string" },
        "postedSlides": { "type": "boolean" },
        "rating": { "type": "number" },
        "tags": {
          "type": "array",
          "items": { "type": "string" },
          "additionalItems": false
        }
      },
      "additionalProperties": false,
      "required": ["firstName", "lastName", "email",
                   "postedSlides", "rating", "tags"
      ]
    }
  },
  "additionalProperties": false,
  "required": ["speaker"]
}
```

Named Object - JSON Doc

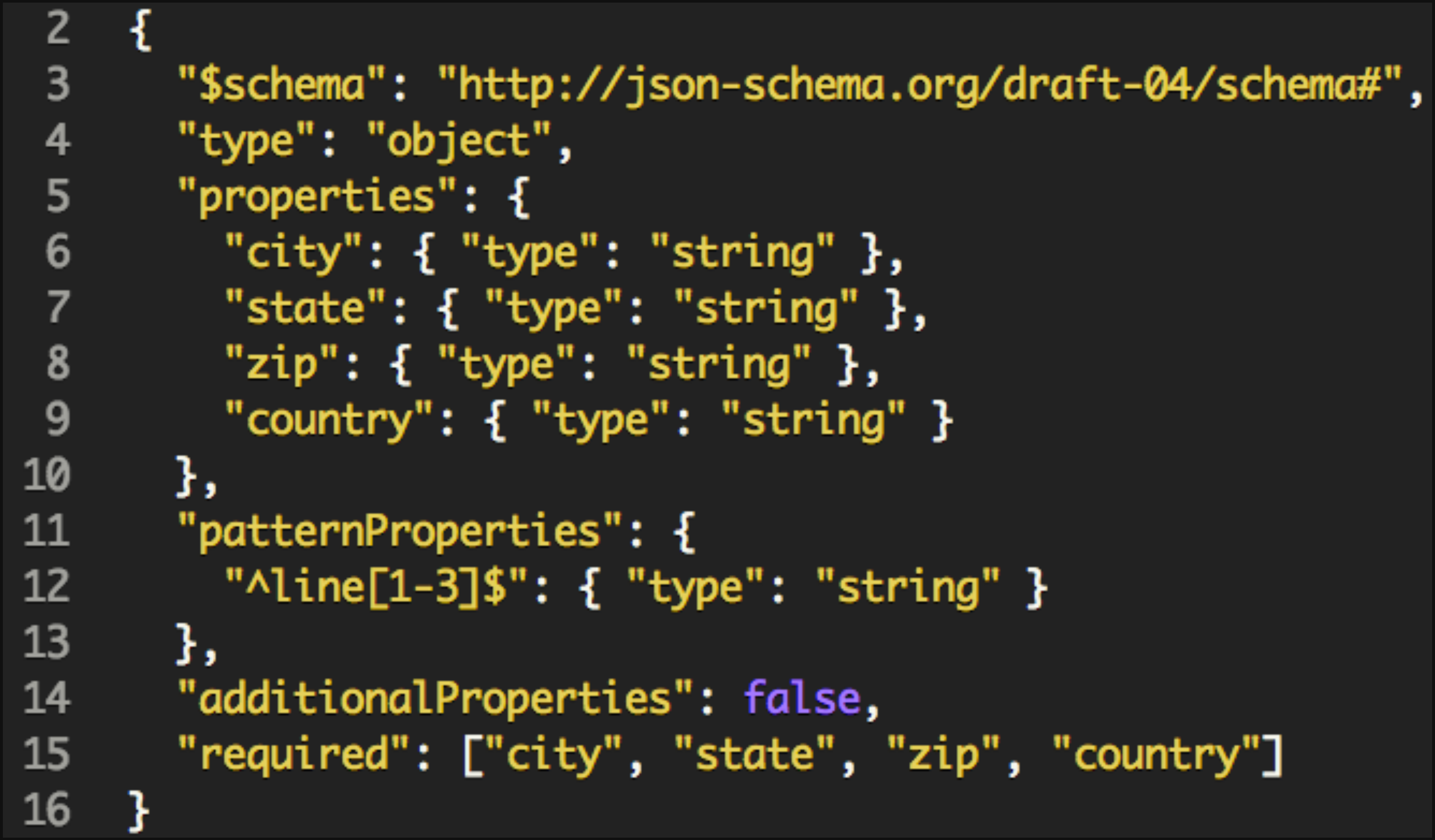


```json
{
  "speaker": {
    "firstName": "Larson",
    "lastName": " Richard",
    "email": "larsonrichard@ecratic.com",
    "postedSlides": true,
    "rating": 4.1,
    "tags": [
      "JavaScript", "AngularJS", "Yeoman"
    ]
  }
}
```

Facets of JSON Schema

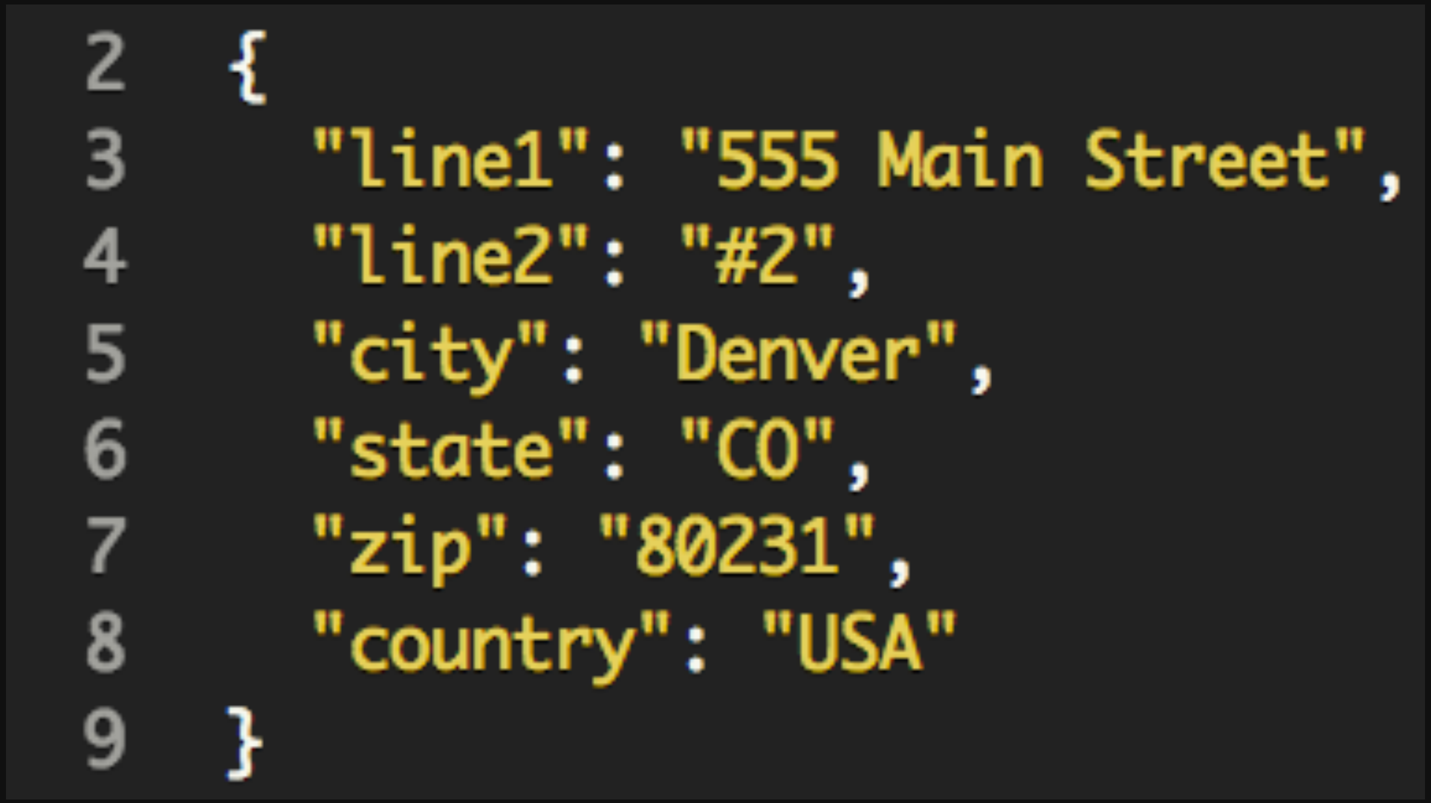


Property Patterns - JSON Schema



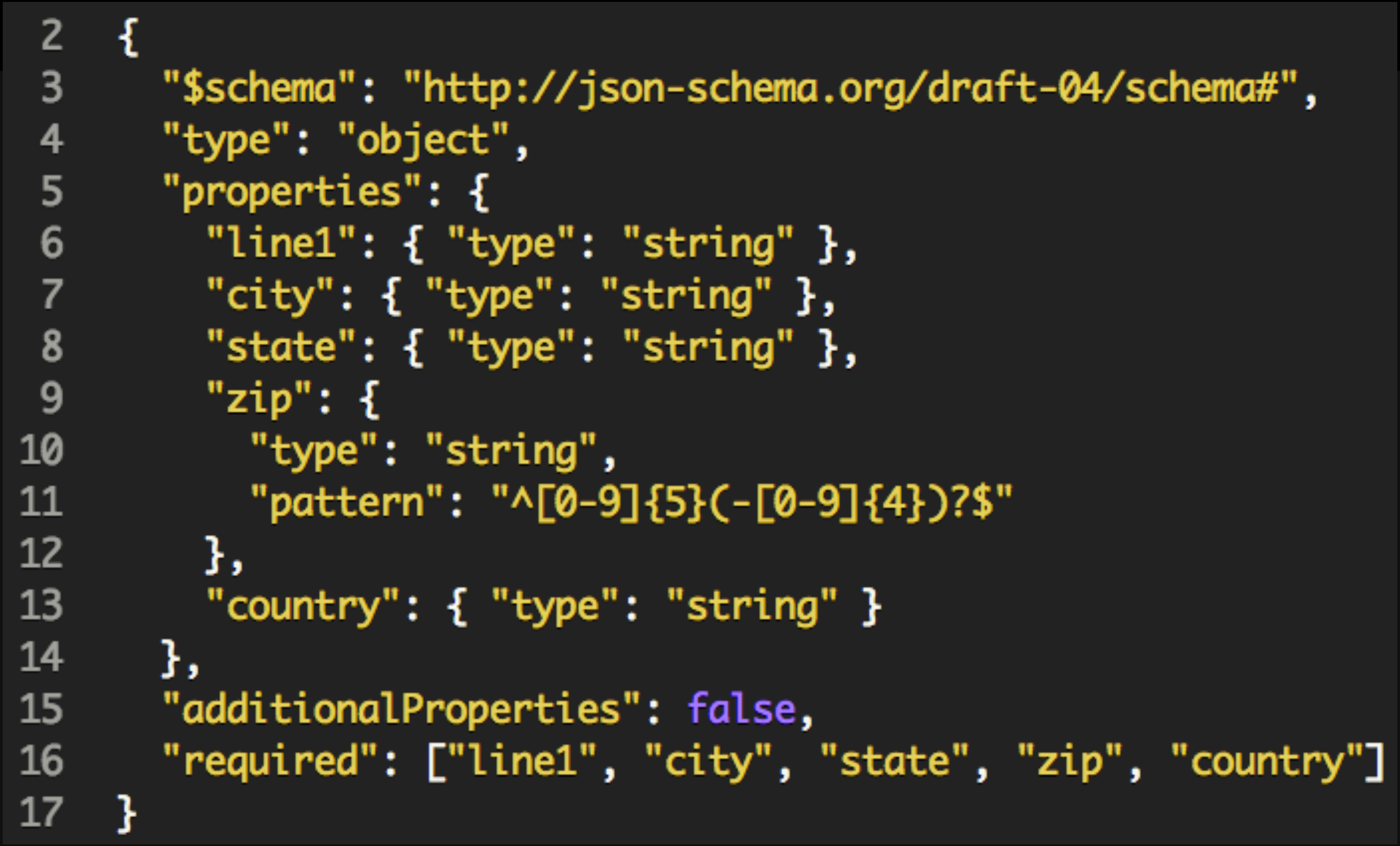
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "city": { "type": "string" },
    "state": { "type": "string" },
    "zip": { "type": "string" },
    "country": { "type": "string" }
  },
  "patternProperties": {
    "^line[1-3]$": { "type": "string" }
  },
  "additionalProperties": false,
  "required": ["city", "state", "zip", "country"]
}
```

Property Patterns - JSON Doc



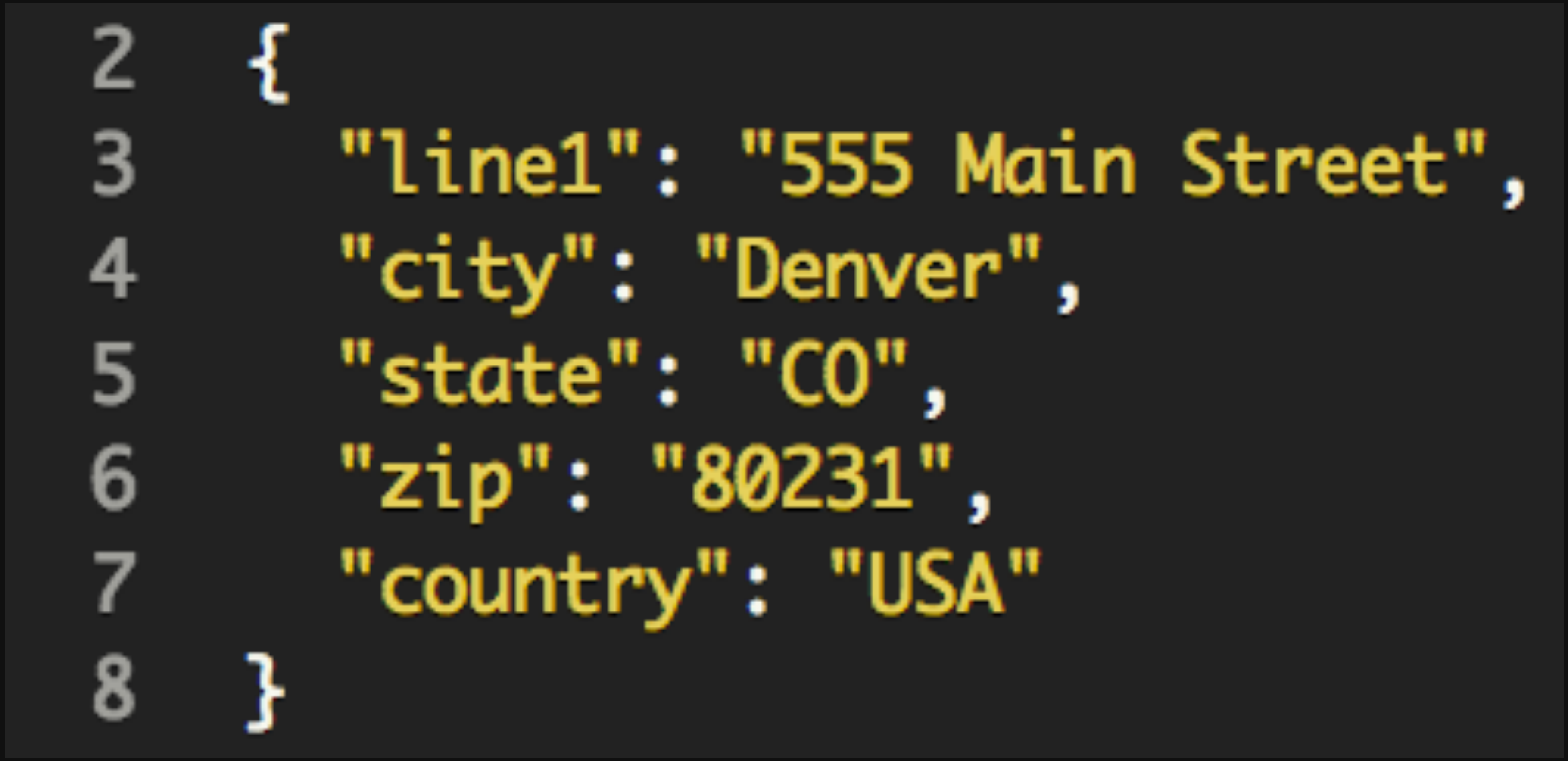
```
{
  "line1": "555 Main Street",
  "line2": "#2",
  "city": "Denver",
  "state": "CO",
  "zip": "80231",
  "country": "USA"
}
```

Regex - JSON Schema



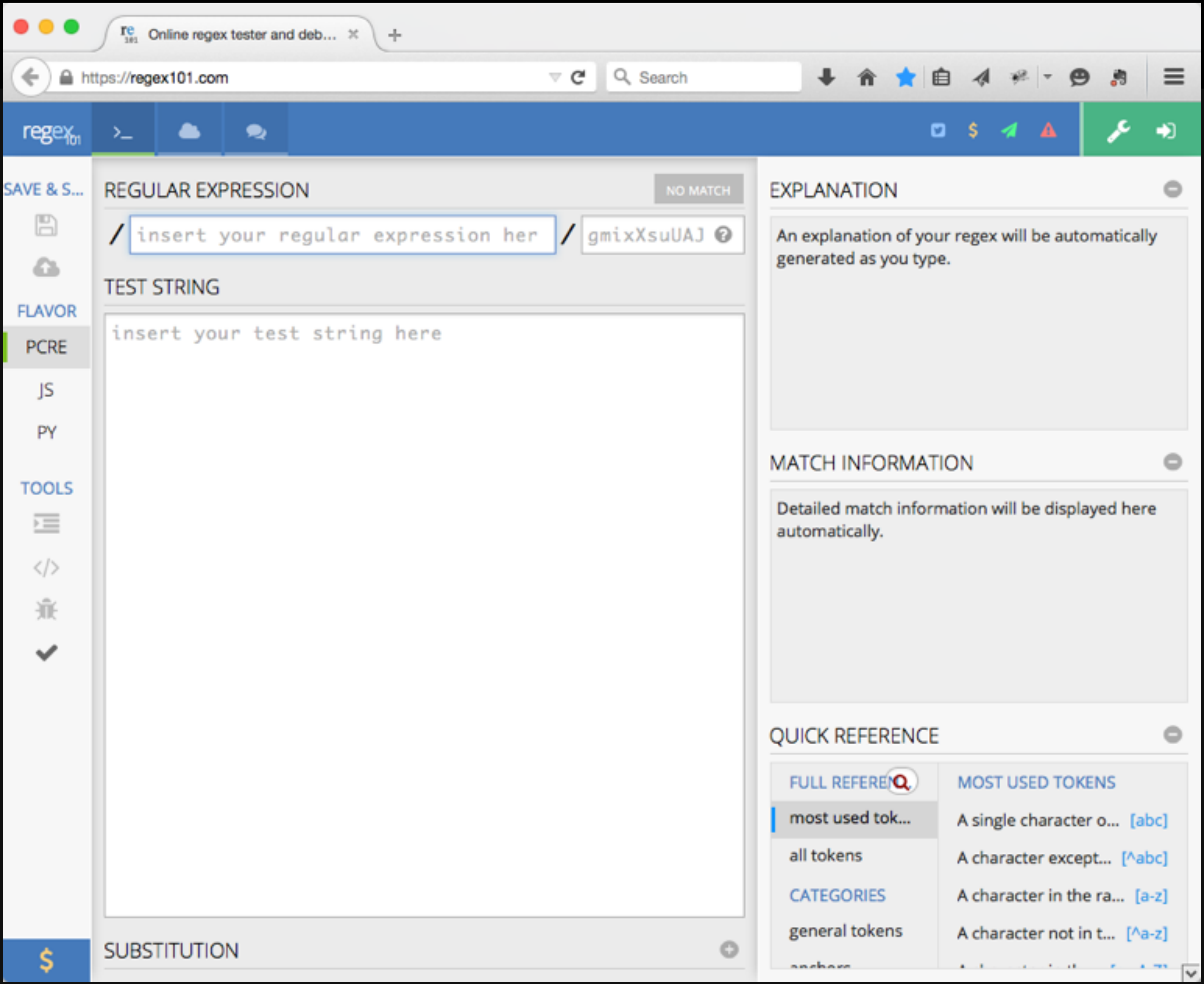
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "line1": { "type": "string" },
    "city": { "type": "string" },
    "state": { "type": "string" },
    "zip": {
      "type": "string",
      "pattern": "^[0-9]{5}(-[0-9]{4})?$"
    },
    "country": { "type": "string" }
  },
  "additionalProperties": false,
  "required": ["line1", "city", "state", "zip", "country"]
}
```

Regex - JSON Doc



```json
{
  "line1": "555 Main Street",
  "city": "Denver",
  "state": "CO",
  "zip": "80231",
  "country": "USA"
}
```

Help!! I'm Awful at Regexp! - Regex101

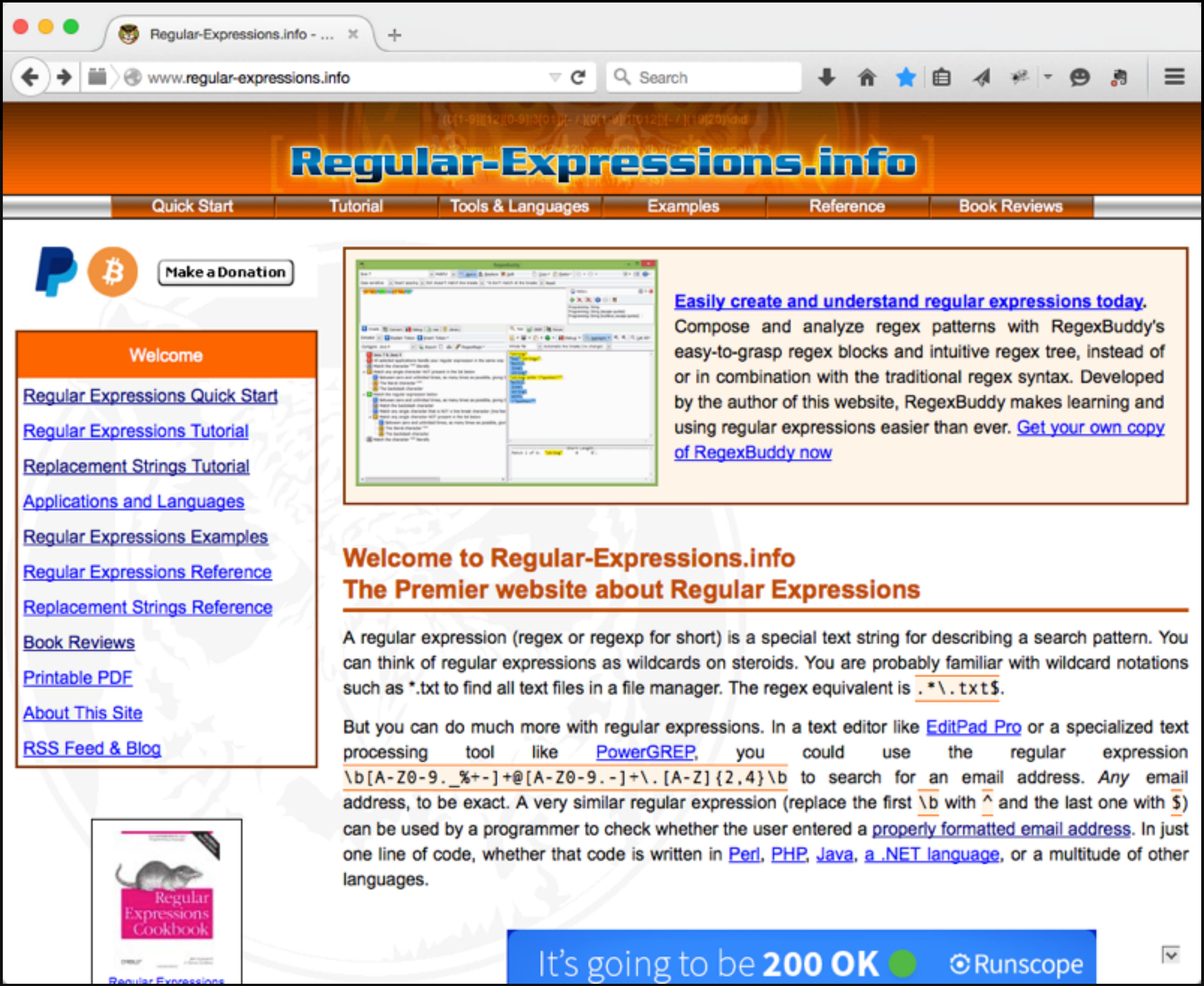


An explanation of your regex will be automatically generated as you type.

Detailed match information will be displayed here automatically.

A single character o...	[abc]
A character except...	[^abc]
A character in the ra...	[a-z]
A character not in t...	[^a-z]

Regular Expressions Info


Regular-Expressions.info - ...
www.regular-expressions.info
Search
Regular-Expressions.info
Quick Start
Tutorial
Tools & Languages
Examples
Reference
Book Reviews
Make a Donation
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
Regular Expressions Cookbook
It's going to be 200 OK
Runscope

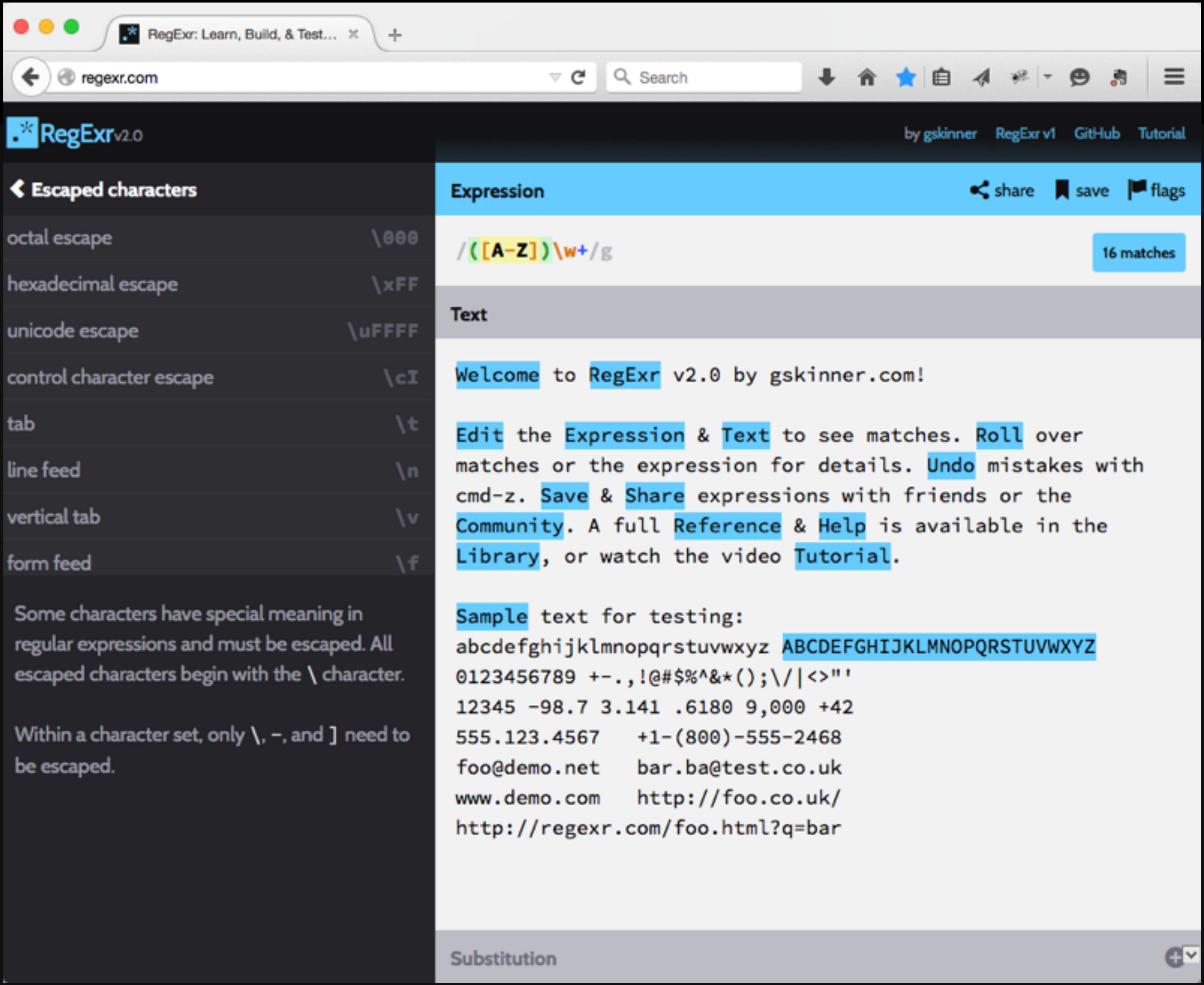
Easily create and understand regular expressions today. Compose and analyze regex patterns with RegexBuddy's easy-to-grasp regex blocks and intuitive regex tree, instead of or in combination with the traditional regex syntax. Developed by the author of this website, RegexBuddy makes learning and using 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.

Regexr



RegExr: Learn, Build, & Test...

regexr.com

RegExr v2.0 — by gskinner · RegExr v1 · GitHub · Tutorial

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 — share · save · flags

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

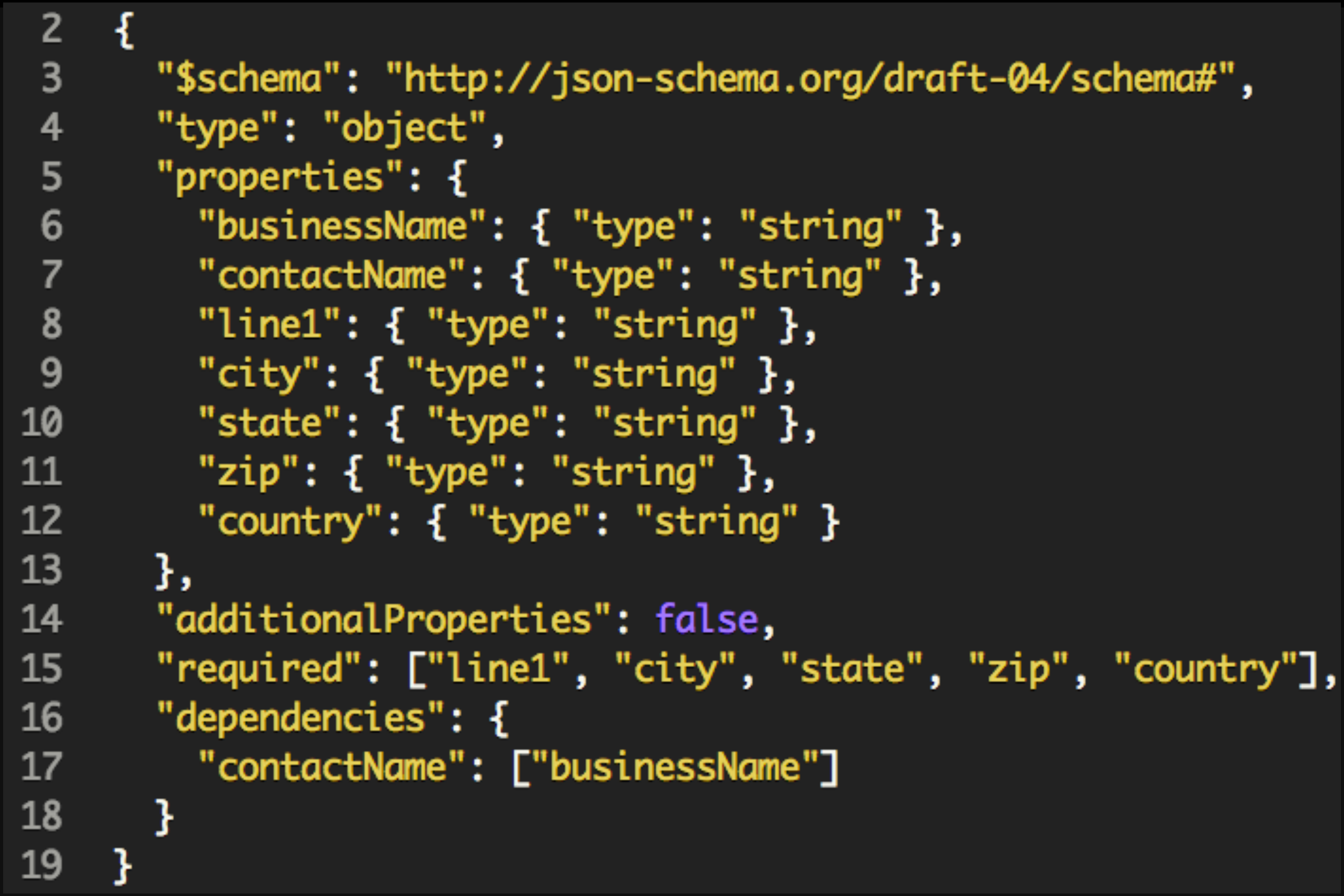
16 matches

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
```

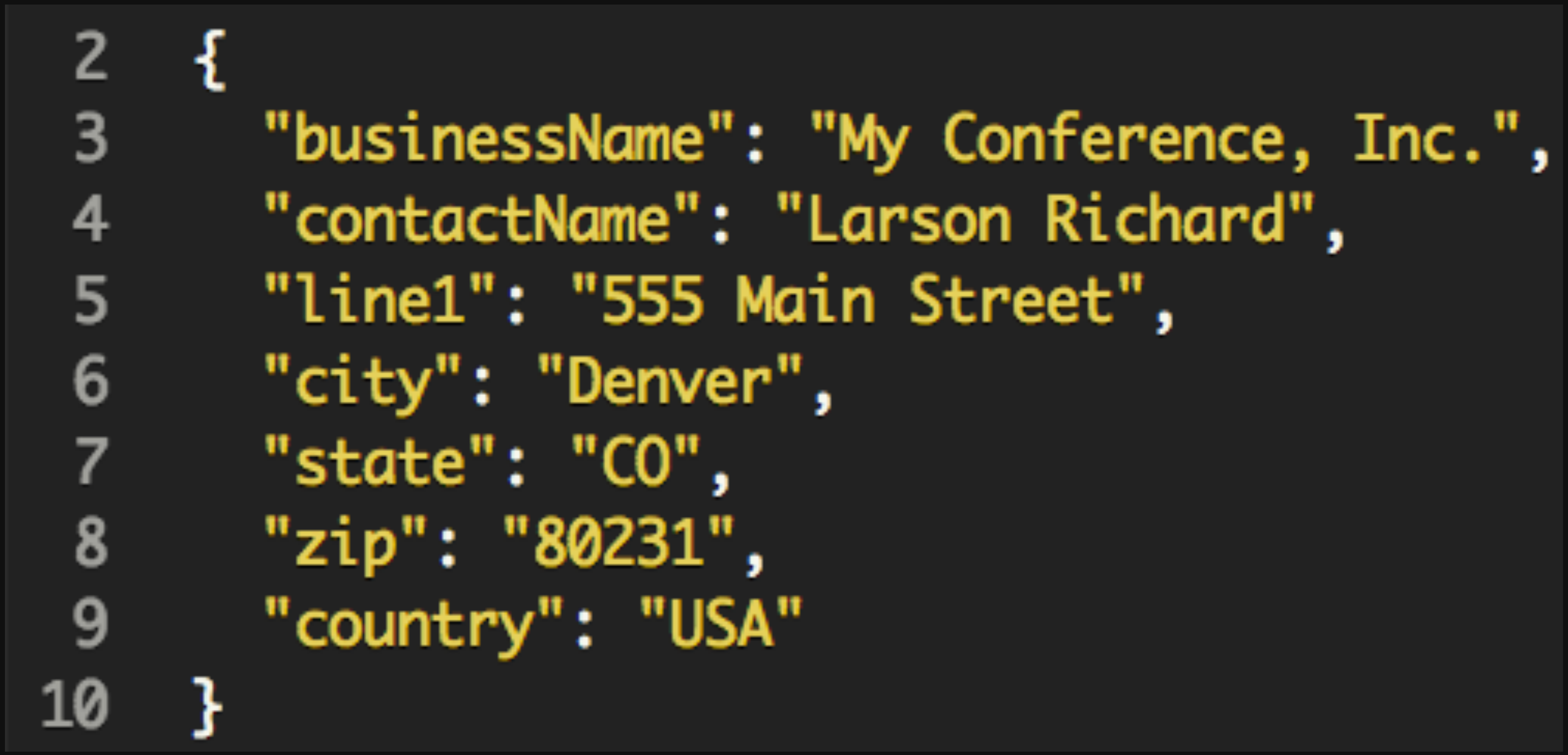
Substitution

Dependent Properties - JSON Schema



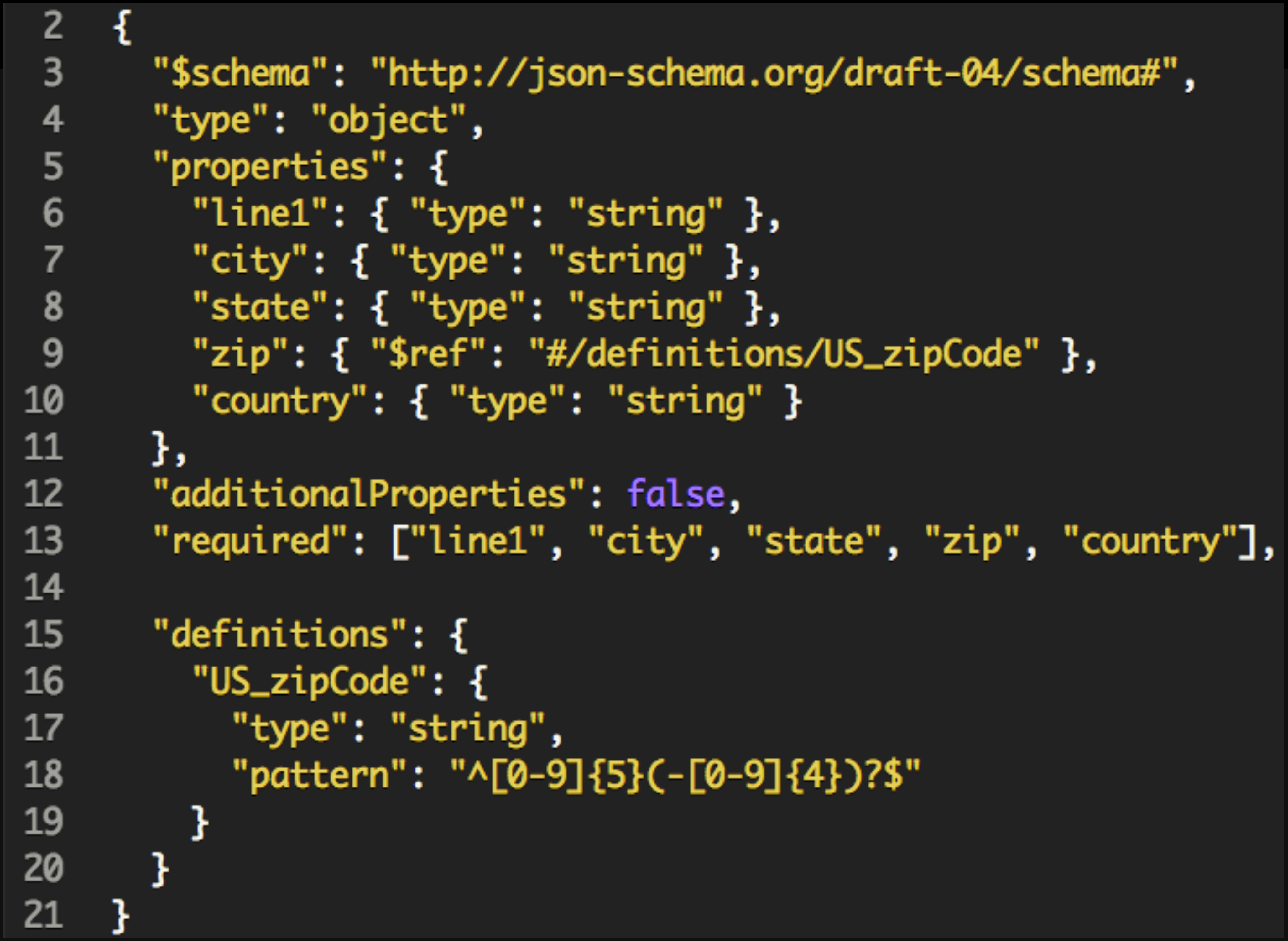
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "businessName": { "type": "string" },
    "contactName": { "type": "string" },
    "line1": { "type": "string" },
    "city": { "type": "string" },
    "state": { "type": "string" },
    "zip": { "type": "string" },
    "country": { "type": "string" }
  },
  "additionalProperties": false,
  "required": ["line1", "city", "state", "zip", "country"],
  "dependencies": {
    "contactName": ["businessName"]
  }
}
```

Dependent Properties - JSON Doc



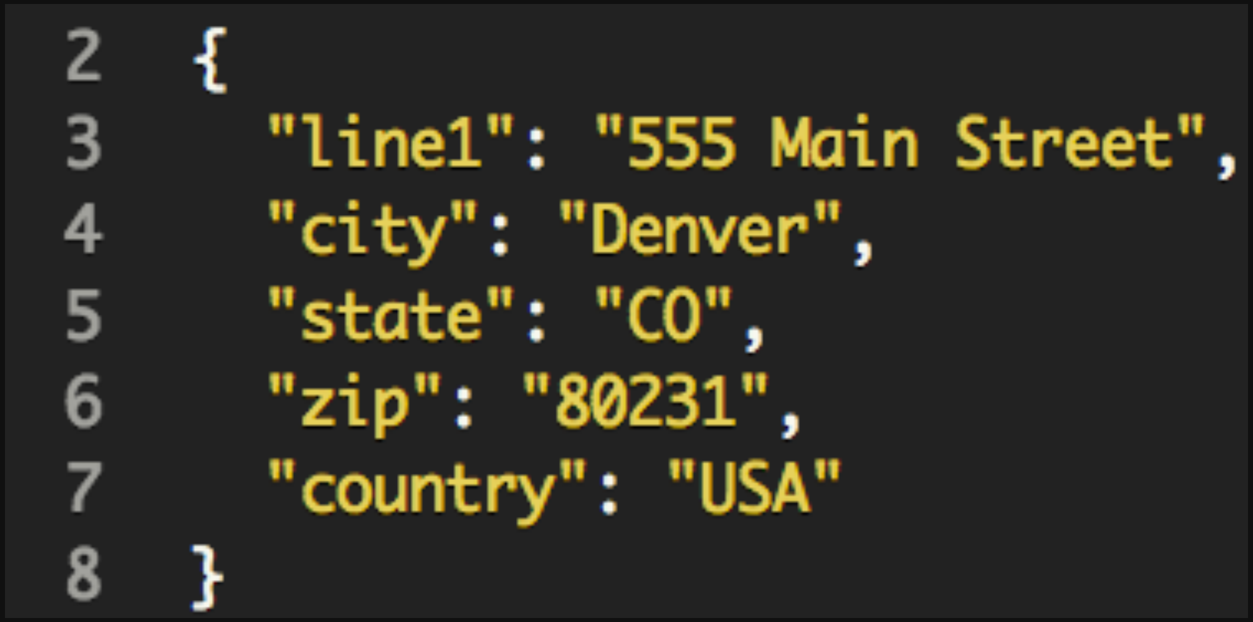
```json
{
  "businessName": "My Conference, Inc.",
  "contactName": "Larson Richard",
  "line1": "555 Main Street",
  "city": "Denver",
  "state": "CO",
  "zip": "80231",
  "country": "USA"
}
```

References (Internal) - JSON Schema



```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "line1": { "type": "string" },
    "city": { "type": "string" },
    "state": { "type": "string" },
    "zip": { "$ref": "#/definitions/US_zipCode" },
    "country": { "type": "string" }
  },
  "additionalProperties": false,
  "required": ["line1", "city", "state", "zip", "country"],

  "definitions": {
    "US_zipCode": {
      "type": "string",
      "pattern": "^[0-9]{5}(-[0-9]{4})?$"
    }
  }
}
```

References (Internal) - JSON Doc



```
{
  "line1": "555 Main Street",
  "city": "Denver",
  "state": "CO",
  "zip": "80231",
  "country": "USA"
}
```

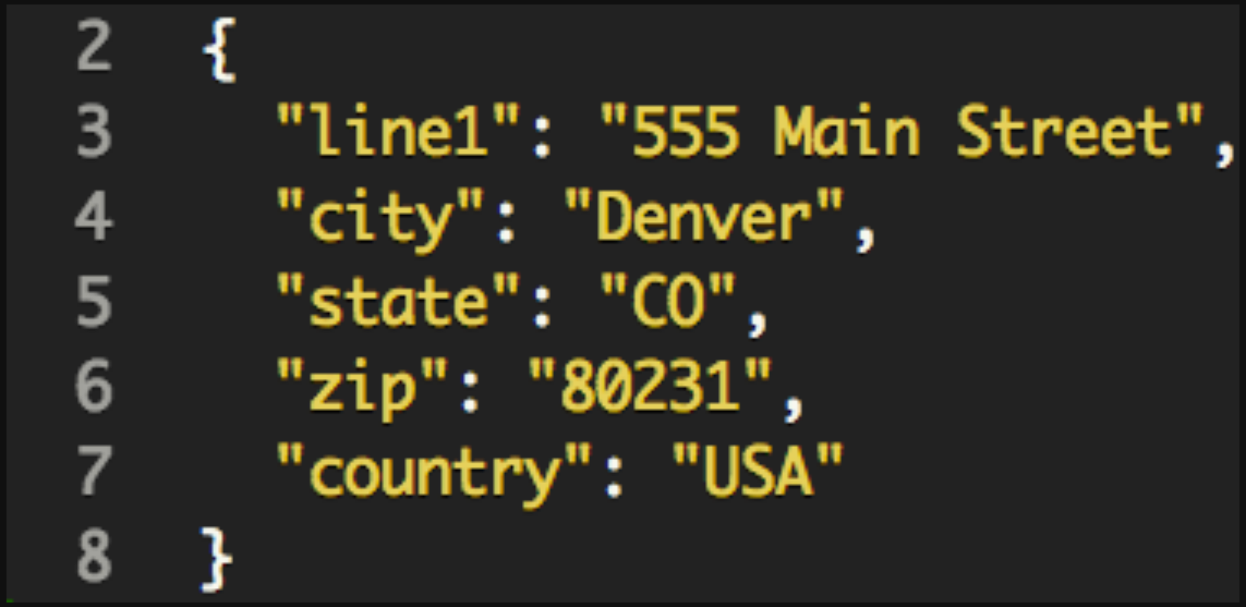
References (External) - JSON Schema

ex-13-external-ref-schema.json

References (Reused) - JSON Schema

ex-13-USCommonAddrSchema.json

References (External) - JSON Doc



```
{
  "line1": "555 Main Street",
  "city": "Denver",
  "state": "CO",
  "zip": "80231",
  "country": "USA"
}
```

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

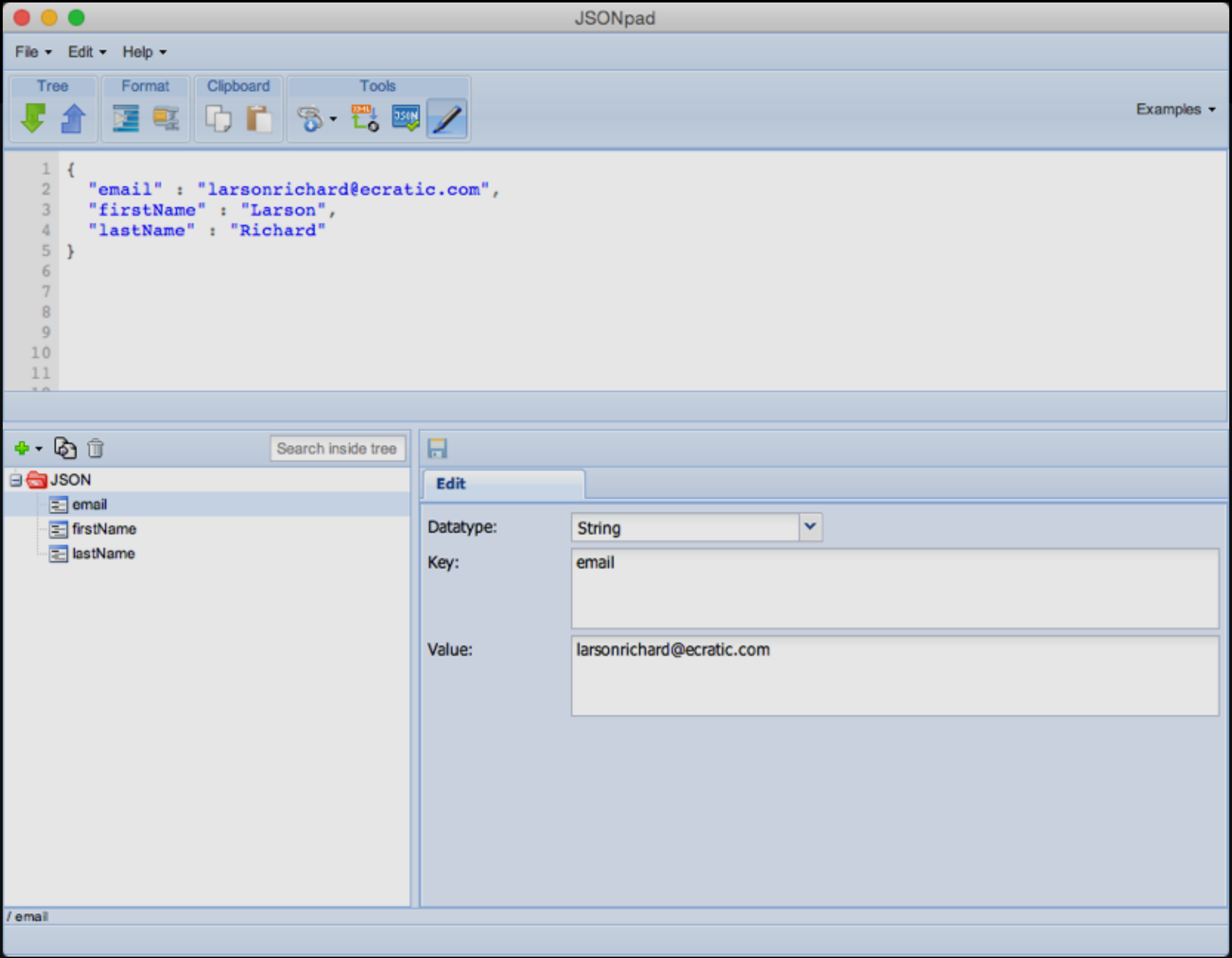
Our Scenario

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

My JSON Schema Workflow for APIs

Model JSON Document	Generate JSON Schema	Validate JSON Document	Document JSON Schema (HTML)
JSONPad https://code.google.com/p/json-pad/	JSON Schema Generator http://jsonschema.net/	JSON Validate http://jsonvalidate.com/	Matic https://github.com/mattyod/matic/
		JSONLint http://jsonlint.com	Docson https://github.com/lbovet/docson

JSONPad Demo



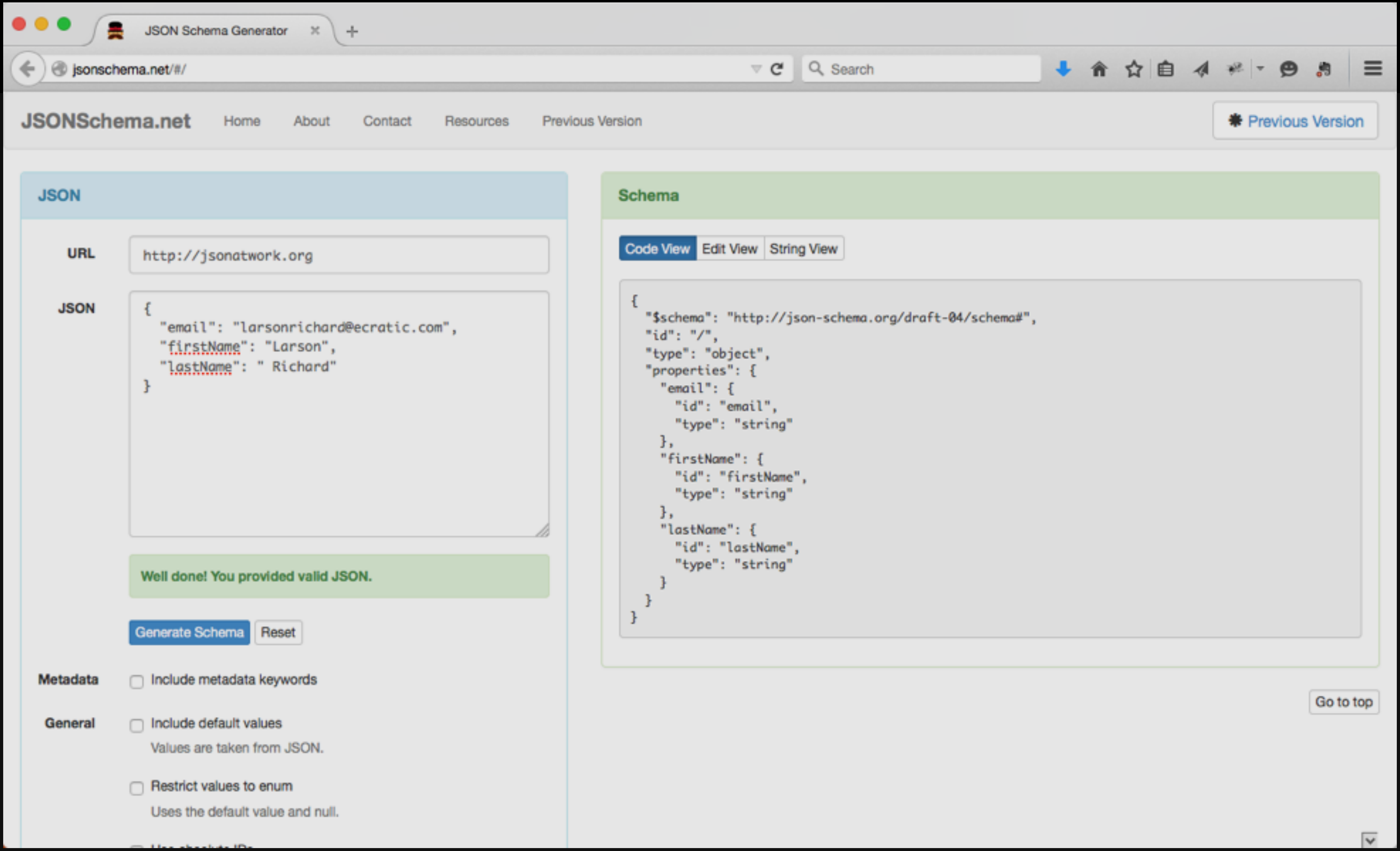
```
{
  "email" : "larsonrichard@ecratic.com",
  "firstName" : "Larson",
  "lastName" : "Richard"
}
```

Datatype: String

Key: email

Value: larsonrichard@ecratic.com

JSON Schema Generator Demo



URL: http://jsonatwork.org

JSON:

```
{
  "email": "larsonrichard@ecratic.com",
  "firstName": "Larson",
  "lastName": " Richard"
}
```

Well done! You provided valid JSON.

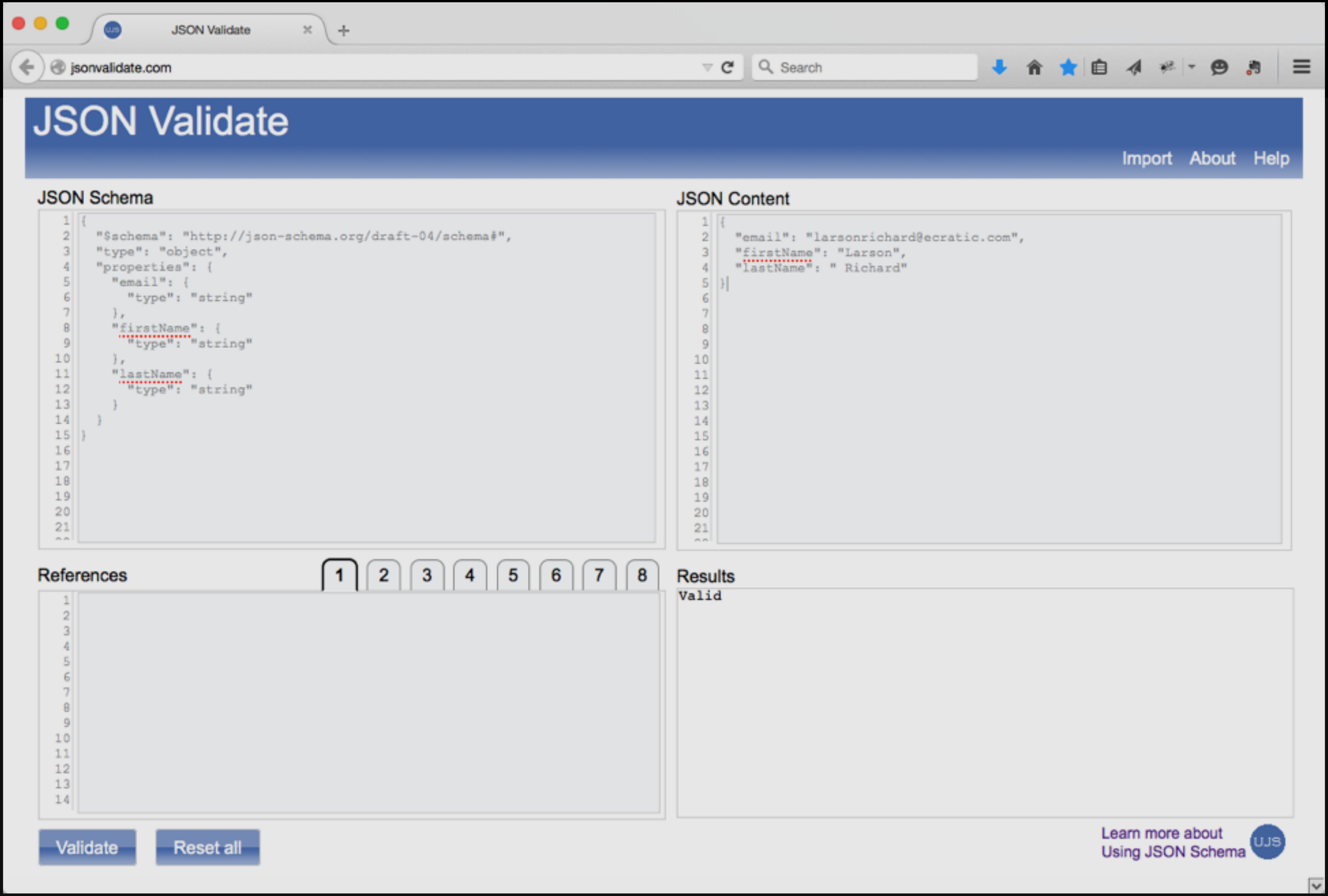
Metadata: ☐ Include metadata keywords

General: ☐ Include default values
Values are taken from JSON.

☐ Restrict values to enum
Uses the default value and null.

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "id": "/",
  "type": "object",
  "properties": {
    "email": {
      "id": "email",
      "type": "string"
    },
    "firstName": {
      "id": "firstName",
      "type": "string"
    },
    "lastName": {
      "id": "lastName",
      "type": "string"
    }
  }
}
```

JSON Validate Demo



JSON Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "email": {
      "type": "string"
    },
    "firstName": {
      "type": "string"
    },
    "lastName": {
      "type": "string"
    }
  }
}
```

JSON Content

```
{
  "email": "larsonrichard@ecratic.com",
  "firstName": "Larson",
  "lastName": " Richard"
}
```

Results

```
Valid
```

Matic

```
npm install -g matic

npm install -g jade


matic
```

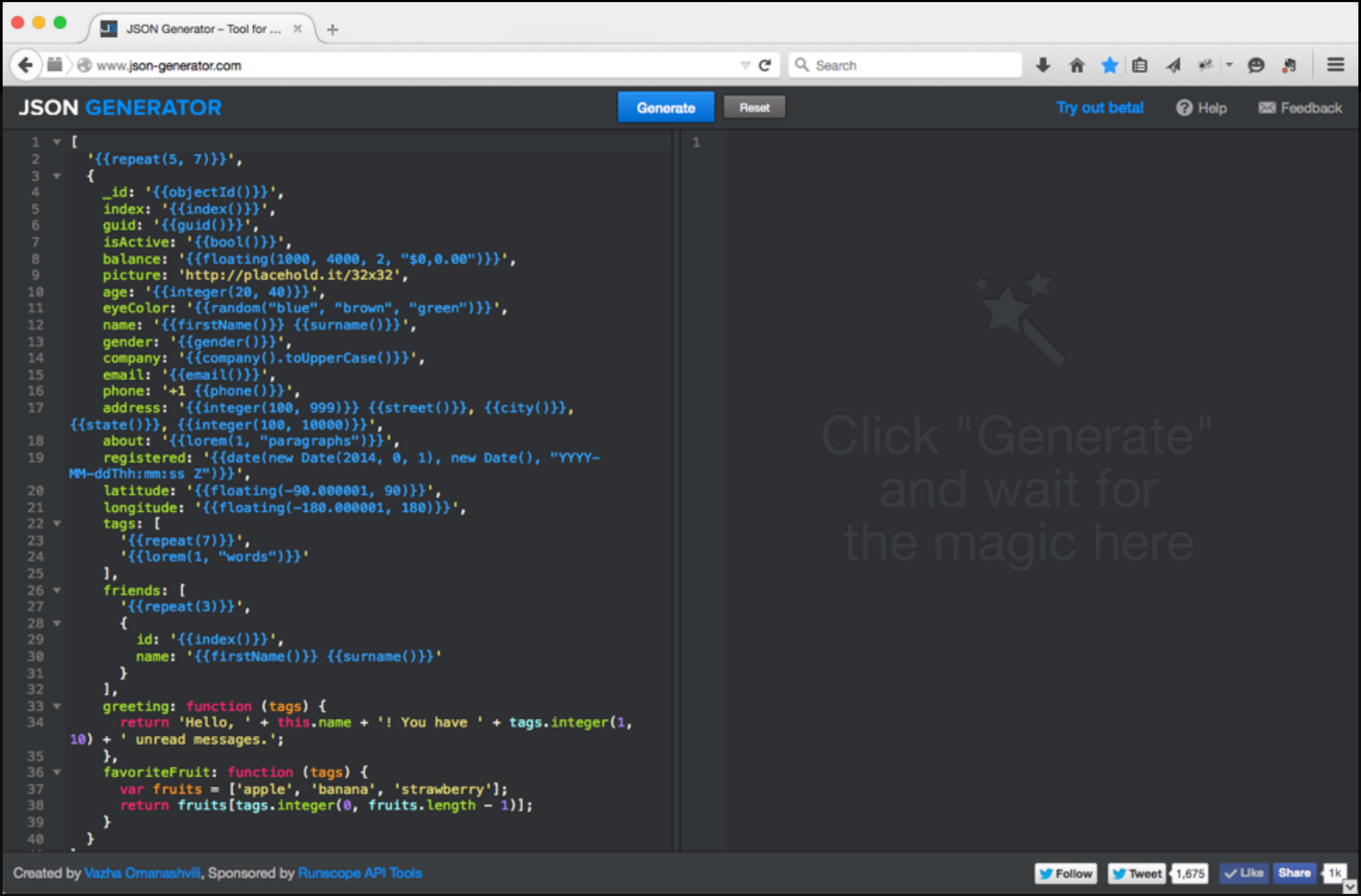
https://github.com/mattyod/matic

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

My JSON API Workflow

Model JSON Document	Generate JSON Document	Deploy Stub REST API
JSONPad https://code.google.com/p/json-pad/	JSON Generator http://www.json-generator.com/	JSON Server https://github.com/typicode/json-server

JSON Generator Demo



```
[
  '{{repeat(5, 7)}}',
  {
    _id: '{{objectId()}}',
    index: '{{index()}}',
    guid: '{{guid()}}',
    isActive: '{{bool()}}',
    balance: '{{floating(1000, 4000, 2, "$0,0.00")}}',
    picture: 'http://placehold.it/32x32',
    age: '{{integer(20, 40)}}',
    eyeColor: '{{random("blue", "brown", "green")}}',
    name: '{{firstName()}} {{surname()}}',
    gender: '{{gender()}}',
    company: '{{company().toUpperCase()}}',
    email: '{{email()}}',
    phone: '+1 {{phone()}}',
    address: '{{integer(100, 999)}} {{street()}}, {{city()}}, {{state()}}, {{integer(100, 10000)}}',
    about: '{{lorem(1, "paragraphs")}}',
    registered: '{{date(new Date(2014, 0, 1), new Date(), "YYYY-MM-ddThh:mm:ss Z")}}',
    latitude: '{{floating(-90.000001, 90)}}',
    longitude: '{{floating(-180.000001, 180)}}',
    tags: [
      '{{repeat(7)}}',
      '{{lorem(1, "words")}}'
    ],
    friends: [
      '{{repeat(3)}}',
      {
        id: '{{index()}}',
        name: '{{firstName()}} {{surname()}}'
      }
    ],
    greeting: function (tags) {
      return 'Hello, ' + this.name + '! You have ' + tags.integer(1, 10) + ' unread messages.';
    },
    favoriteFruit: function (tags) {
      var fruits = ['apple', 'banana', 'strawberry'];
      return fruits[tags.integer(0, fruits.length - 1)];
    }
  }
```

Click "Generate" and wait for the magic here

Created by Vazha Omanashvili, Sponsored by Runscope API Tools

JSON Server

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

```
json-server speakers-data.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

livingsocial

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