JSON Schema for HTTP API-s

@ikr

JSZurich + Webtuesday lightning talk

14.01.2014

Premise

“The secret to building large apps is NEVER build large apps. Break up your applications into small pieces. Then, assemble those testable, bite-sized pieces into your big application.”

–Justin Meyer

The kind of “pieces” defines a software architecture style

- How the *problem* is *decomposed*
- How the *solution* is *composed*

```
LET piece := software component
```

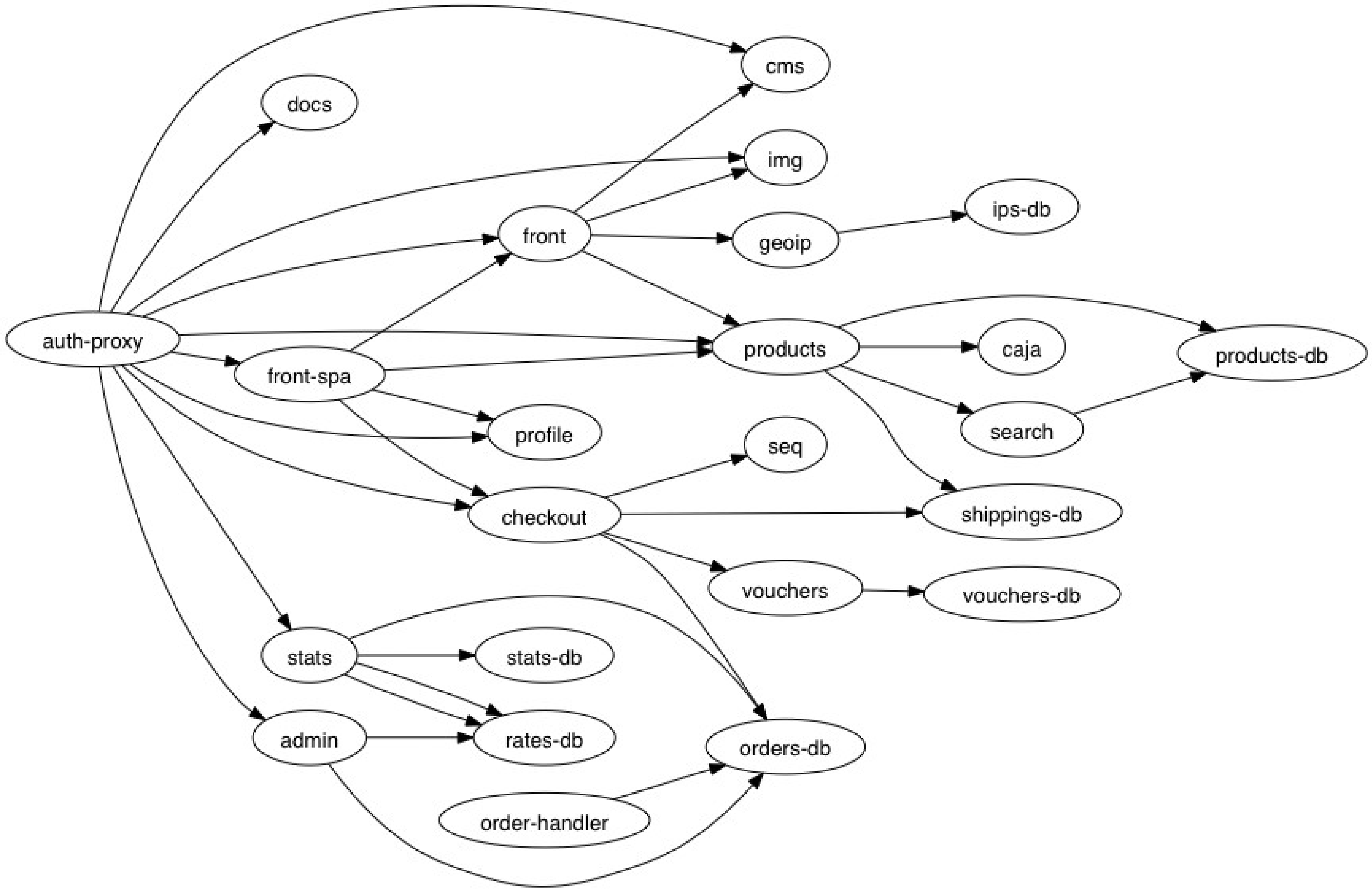
Component

- Unit of software deployment
- A separate active process

Not component

- Programming language namespace or class
- File
- In-process linked library: jar, dll, composer- or npm package

Components as if you mean it



Components' dependencies

Benefits

- HTTP
- Super-low coupling
- Understandability of parts
- Small and fast test suites
- Accessibility: just `curl` and `jsontool`
- Scalability
- Mostly stateless
- Value-based and data-oriented
- Composability via pipes and filters
- Whole new feature in a few lines of bash

Drawbacks: some questions become harder to answer

- I'm GET-ting a list of entities. What properties do they have?
- I need to change that entity's structure. What's affected?

JSON Schema to the rescue

Describes the JSON format of your resources/entities *in JSON*

IETF draft

```
"currency": {
    "type": {
        "enum": [
            "CHF", "EUR",
            "GBP", "RUB", "USD"
        ]
    }
}

"cost": {
    "type": "object",

    "properties": {
        "currency": {"$ref": "#currency"},

        "amount": {
            "type": "string",
            "pattern": "^[0-9]+\\.[0-9]{2}$"
        }
    },

    "required": ["currency", "amount"]
}
```

```
"destination": {
    "type": "object",

    "oneOf": [
        {"$ref": "#destinationCity"},
        {"$ref": "#destinationRegion"},
        {"$ref": "#destinationHotel"}
    ]
},

"periodOfStay": {
    "type": "object",

    "properties": {
        "checkInDate": {
            "type": "string", "format": "date"
        },

        "checkOutDate": {
            "type": "string", "format": "date"
        }
    },

    "required": ["checkInDate", "checkOutDate"]
},
```

JSON Schema is a soft type system for your application's data flow

```
{"currency": "USD", "amount": "1000000000.00"}

{"destination": {"hotelId": "13143"}}

{
    "checkInDate": "2014-01-14",
    "checkOutDate": "2014-01-15"
}
```

Exposing via HTTP

- Schema has a URL
- Format declaration in header

```
Link: <http://example.com/my-hyper-schema#>;
rel="describedBy"
```

- JSON Hyper-Schema
- JSON+HAL

Automatic verification

- Verifying middleare for requests and responses
- Even HTML forms' validation
- May be enabled only in development mode
- Profit!