Real World REST - Beginners

International PHP Conference 2015

Tobias Schlitt / @tobySen

2015-06-09

passion for software quality

international PHP2015 conference - spring edition -

What is REST?

What do you think is REST?

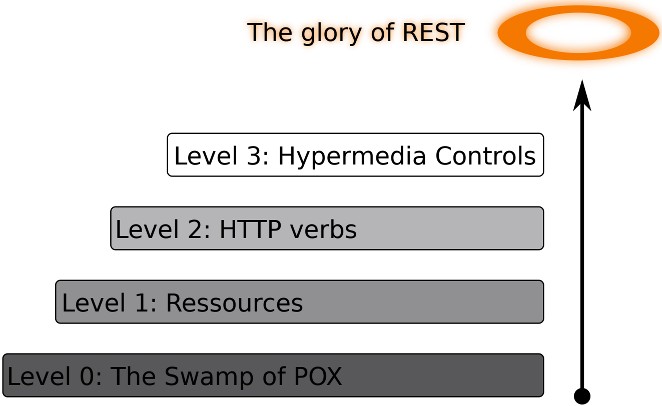
Architecture style for distributed systems

* Abstraction of the web

Goals

- Scalably
- Simple
- Modifiability
- Uniform interface
- Re-use existing . . .
  - concepts
  - infrastructure
  - components

RESTfulness



The glory of REST

Level 3: Hypermedia Controls

Level 2: HTTP verbs

Level 1: Ressources

Level 0: The Swamp of POX

Source: http://martinfowler.com/articles/richardsonMaturityModel.html

REST is not HTTP

- Developed paralell to HTTP 1.1
- Not necessarily HTTP based
- But most commonly

The Concepts

- Resources
- Methods
- Relations

Outline

Resources

Methods

Media Types

Resource Relations

Error Handling

Resources

- Entities or collections
- Tree structure
- URI as **unique** identifier
- Examples:
  - `https://plan.qafoo.com/users`
  - `/users/toby`
  - `//jobs/23`
  - `../`

Outline

Resources

Methods

Media Types

Resource Relations

Error Handling

HTTP Methods

- GET
- HEAD
- OPTIONS
- TRACE
- POST
- PUT
- DELETE

Safe Must not change state

Idempotent Repeating must result in same state

passion for software quality

Examples: Read

- → `GET /users/kore`
- ← Representation of the user

- → `HEAD /jobs/23`
- ← Only meta data of the job representation

passion for software quality

Examples: Write

- → `PUT /users/kore`
- **Replaces** the user entity **completely**

- → `POST /jobs`
- **Creates** a new entity **inside** `/jobs`

Additional Methods

PATCH Partial updates (RFC 5789)

COPY Create a copy of a resource (RFC 2518)

MOVE Move a resource (RFC 2518)

...

Custom Methods

- HTTP allows server to support custom methods
- Semantics must be carefully designed
  - Use HTTP properties
  - Try especially Idempotence

Outline

Resources

Methods

Media Types

Resource Relations

Error Handling

State Transfer

- REST is about state
  - Retrieve
  - Manipulate
  - Drive workflow
- How does a client interprete state?

Media Types

- Well known media types
  - `text/html`
  - `image/png`
  - `application/vnd.adobe.flash-movie`
  - `application/vnd.apple.installer+xml`
- Media type assigns semantic to raw data
- `http://qa.fo/iana-types`

Media Type Schematics

```
type / [tree.]  subtype [+suffix] [; parameters]
```

- `application/psr.com.qafoo.plan-job+json`
- `application/psr.com.qafoo.plan-job+xml; charset=UTF-8`

Media Type Structure

- Use standard base structure
  - XML
    - More powerful
    - More complex to consume
  - JSON
    - Reduced feature set
    - Easy to consume
- Ensure UTF-8

Structure Examples

```
{
    "title": "Design␣a␣REST␣API",
    "minDays": 3,
    "maxDays": 5,
    "assignees": [
        "http://time.qafoo.com/users/toby"
    ],
}
```

```
<?xml version="1.0"?>
<job xmlns="urn:psr.com.qafoo.plan-job">
    <title>Design a REST API</title>
    <minDays>3</minDays>
    <maxDays>5</maxDays>
    <assignee>
        http://time.qafoo.com/users/toby
    </assignee>
</job>
```

Media Type Headers

- `Content-Type`
- `Content-Length`
- `Accept`

Outline

Resources

Methods

Media Types

Resource Relations

Error Handling

Relations

- Hyperlinks!
- Drive the workflow
  - HATEOAS (advanced session)

Link Standards

- Several Standards for links:
  - X-Link
  - Atom
  - …
- Stick to a standard

passion for software quality

Links in Representations

```xml
<?xml version="1.0"?>
<job xmlns="urn:psr.com.qafoo.plan-job"
    xmlns:atom="http://www.w3.org/2005/Atom">
    <atom:link rel="self"
        type="application/psr.com.qafoo.plan-job+xml"
        href="/jobs/23" />
    <atom:link rel="collection"
        type="application/psr.com.qafoo.plan-job-list+xml"
        href="/jobs" />
    <!-- ... -->
</job>
```

http://qa.fo/iana-relations

Qafoo
passion for software quality

Outline

Resources

Methods

Media Types

Resource Relations

Error Handling

Indicating Error

- Error status codes
  - `4xx` Client error
  - `5xx` Server error
- Include explaining body
  - Fit media type structure

Bottom Line

- REST = Architecture Type
- Abstraction of the web
- Basic concepts:
  - Resources
  - URIs
  - Methods (verbs)
  - Media Types
  - Hyperlinks
- Questions?

Join the Advanced Session

- Caching
- Authentication
- Content Negotiation
- Versioning
- **How much REST do you need?**

My project by heart

Smart PHP Timeline Profiler

`http://tideways.io`

30% for 3 month: BERLIN15

https://joind.in/talk/view/13504

passion for software quality

THANK YOU

Rent a quality expert
qafoo.com