DDD & REST

Domain-Driven APIs for the web

Oliver Gierke

/twitter/olivergierke
Oliver Gierke

Pivotal (formerly SpringSource), Dresden, Germany

Followers 555, Starred 55, Following 31

Contributions

Organizations

Repositories contributed to

Repositories

Contributions

Summary of pull requests, issues opened, and commits. Learn how we count contributions.

Learn more
Background
Spring Data REST

Spring Data
Repositories & Aggregates

Spring HATEOAS
Hypermedia for Spring MVC
REST ≠ CRUD via HTTP
“What does it take to bridge the worlds of DDD & REST?”
Value objects
Value Objects are a PITA to build in some languages.
Still, they’re worth it.

See „Power Use of Value Objects in DDD“ by Dan Bergh Johnsson.
Lombok — putting the spice back into Java.
Key opponents:
Mapping libraries that need to (de)serialize them.
Entities & Repositories
Entity + Repository = Aggregate
Aggregates form nice representation boundaries.
Aggregates become the key things to refer to.
Don’t get trapped by datastore thinking.
Try to avoid bi-directional relationships.
Domain Events
Level 0: No events at all
Level 0: No events at all

Level 1: Explicit operations
If you’re calling two setters in a row, you’re missing a concept.
Level 0: No events at all

Level 1: Explicit operations

Level 2: Some operations as events
Domain events as state transitions.
Expose important events to interested parties via feeds.
Level 0: No events at all

Level 1: Explicit operations

Level 2: Some operations as events

Level 3: Event Sourcing
REST
Representation design matters
Aggregates

Identifiable
Referable
Scope of consistency
Resources

Identifiable
Referable
Scope of consistency
Hypermedia
Serving data and navigation information at the same time.
Trading domain knowledge with protocol complexity in clients.
Reducing decisions in clients to whether a link is present or not.
Prefer explicit state transitions over poking at your resources using PATCH.
Translate domain concepts into web-appropriate ones.
Aggregate Root / Repository

Relations

IDs

@Version

Last Modified Property

Collection / Item Resources

Links

URIs

ETags

Last Modified Header
RESTBucks
RESTBucks

1. payment expected
2. preparing
3. cancelled
4. ready
5. completed
<table>
<thead>
<tr>
<th>Method</th>
<th>URI</th>
<th>Action</th>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td>/orders</td>
<td>Create new order</td>
<td>1</td>
</tr>
<tr>
<td>POST/PATCH</td>
<td>/orders/{id}</td>
<td>Update the order</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(only if &quot;payment expected&quot;)</td>
<td></td>
</tr>
<tr>
<td>DELETE</td>
<td>/orders/{id}</td>
<td>Cancel order</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(only if &quot;payment expected&quot;)</td>
<td></td>
</tr>
<tr>
<td>PUT</td>
<td>/orders/{id}/payment</td>
<td>Pay order</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(only if &quot;payment expected&quot;)</td>
<td></td>
</tr>
<tr>
<td>GET</td>
<td>/orders/{id}</td>
<td>Poll order state</td>
<td>5</td>
</tr>
<tr>
<td>GET</td>
<td>/orders/{id}/receipt</td>
<td>Access receipt</td>
<td></td>
</tr>
<tr>
<td>DELETE</td>
<td>/orders/{id}/receipt</td>
<td>Conclude the order process</td>
<td>6</td>
</tr>
</tbody>
</table>

Barista preparing the order

<table>
<thead>
<tr>
<th>Method</th>
<th>URI</th>
<th>Action</th>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>/orders/{id}</td>
<td>Poll order state</td>
<td>5</td>
</tr>
<tr>
<td>GET</td>
<td>/orders/{id}/receipt</td>
<td>Access receipt</td>
<td></td>
</tr>
<tr>
<td>DELETE</td>
<td>/orders/{id}/receipt</td>
<td>Conclude the order process</td>
<td>6</td>
</tr>
<tr>
<td>Method</td>
<td>Resource type</td>
<td>Action</td>
<td>Step</td>
</tr>
<tr>
<td>------------</td>
<td>---------------</td>
<td>--------------------------</td>
<td>------</td>
</tr>
<tr>
<td>POST</td>
<td>orders</td>
<td>Create new order</td>
<td>1</td>
</tr>
<tr>
<td>POST/PATCH</td>
<td>update</td>
<td>Update the order</td>
<td>2</td>
</tr>
<tr>
<td>DELETE</td>
<td>cancel</td>
<td>Cancel order</td>
<td>3</td>
</tr>
<tr>
<td>PUT</td>
<td>payment</td>
<td>Pay order</td>
<td>4</td>
</tr>
</tbody>
</table>

**Barista preparing the order**

<table>
<thead>
<tr>
<th>Method</th>
<th>Resource type</th>
<th>Action</th>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>order</td>
<td>Poll order state</td>
<td>5</td>
</tr>
<tr>
<td>GET</td>
<td>receipt</td>
<td>Access receipt</td>
<td></td>
</tr>
<tr>
<td>DELETE</td>
<td>receipt</td>
<td>Conclude the order process</td>
<td>6</td>
</tr>
</tbody>
</table>
Spring
RESTBucks
<table>
<thead>
<tr>
<th></th>
<th>Orders</th>
<th>Payment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Web</strong></td>
<td>Spring Data REST</td>
<td>Manual implementation</td>
</tr>
<tr>
<td><strong>Service</strong></td>
<td>-</td>
<td>Manual implementation</td>
</tr>
<tr>
<td><strong>Repository</strong></td>
<td>Spring Data</td>
<td>Spring Data</td>
</tr>
</tbody>
</table>
JacksonCustomizations
Externalize tweaks to the general JSON design
Spring Data REST for the CRUDdy parts.
ResourceProcessor
To conditionally sneak links into the default representation.
Code

Spring RESTBucks - https://github.com/olivergierke/spring-restbucks
Questions?