Advanced Topics

Objectives

- Give an overview of the expected idempotency of GET requests, and the consequences of this
- Give an overview of techniques that can mitigate problems arising from repeating intrinsically non-idempotent requests (for example, credit card payments) such as might arise during network partial failure conditions
- Give an overview of potential benefits of client-side, or near-client, caching of data, and how the E-Tag technique can facilitate this
- Give an overview of the HATEOAS concept, and why Intuit does not favor its use

Objectives

- Give an overview of how to define a new JAX-RS annotation for a non-standard HTTP method
- Give an overview of the purpose of the IETF proposed “PATCH” HTTP method, the general form of a JSON Patch document, the potential benefit to client and near-client caches of PATCH over using PUT

Idempotency

- Ensure an operation's effect is "only once", even if the operation is performed multiple times.
  - HTTP specification says GET operations should only read, therefore, multiple calls do not change anything
- For other operations, this is more difficult
  - Mostly, HTTP specification allows non-idempotent behavior
  - Networks exhibit "partial failure"—a request might not get a response, but how can we know if the request was lost, or if it was acted on and the response was lost?

Handling Partial Failure

- Many operations in Intuit services must be idempotent in the face of partial failure
  - Credit card payment
- Any particular case might have unique requirements
- In general, an operation ID might be used
  - Client creates a sequence number for a request
  - Repeated requests carry the same sequence number
  - Server records (perhaps temporarily) the sequence number and client identification (IP, perhaps?) and ignores repeated requests

Caching, Performance, Liveness

- Repeated requests are bad for performance and scalability
  - Waste CPU/Disk/Network bandwidth
- Caching can avoid requests entirely
  - But results in potentially stale data
- ETags can be used to identify the “version” of a data set
  - Conditional requests can be sent indicating “I have this version, send only if changed”
  - Network round trip still happens, but computation and bandwidth can often be greatly reduced

HATEOAS

- Hypermedia As The Engine Of Application State
  - Proposed by Roy Fielding PhD. to allow client software to handle change and extension in the server
  - In effect, server specifies legal logic for client in hypermedia (linked) document, rather than client code
    - Compare with web pages that present users with options of what to do next
  - Theoretically supports substantial changes on the server without breaking the client

HATEOAS

- Much harder to implement for machine clients than human clients, in real systems
- Machine readable hypermedia significantly raises the barrier to entry
  - Might be why it’s pretty rare
- “Hypermedia” isn’t required to be HTML, might be some other structure (e.g. JSON objects with links in them, or perhaps BPCL-like)
- Might tend to lock clients into “presentation half of application” rather than being independent applications built over a general service

What Is PATCH For?

- According to the HTTP specification PUT overwrites the entire resource
  - This requires the entre object to be sent, even if only one field is to be updated
  - It keeps the data set more consistent
  - It allows local caches (e.g. proxy servers) to know the state of a resouce
    - This is probably not useful in the case of concurrent updates to the resource from different clients (c.f. ETags)
- There are three approaches to mitigating this of interest at Intuit

What Is PATCH For?

- Use a PUT operation, but arrange that null (missing) fields are left unchanged
  - Breaks the PUT specification, but is commonly used, and has been standard at Intuit for some time
  - Cannot express “make this field null”!
- Use a PUT operation, but use the “fields” query parameter (formerly “elements”) to indicate which fields to be updated
  - Same as previous, but allows “assignment to null” problem

What Is PATCH For?

- Use a PATCH operation
  - Proposed HTTP standard for partial updates
  - Accepted as an Intuit standard 2015
- Two approaches
  - Incomplete object, as PUT but doesn’t lie to caches
  - JSON Patch, use a special JSON document indicating what changes should be made to what fields
    - JSON Patch documents are of MIME type application/json-patch+json
    - JSON Patch documents are an array of “operations” such as add, remove, replace, move, copy
    - Operations form an atomic set of updates, succeeding or failing as a group

Using PATCH In JAX-RS

- JAX-RS does not provide @PATCH annotation, but this can easily be added:

```
@Target(ElementType.METHOD)
@Retention(RetentionPolicy.RUNTIME)
@HttpMethod("PATCH")
public @interface PATCH { }
```