JMAP for Migration and Data Portability

IETF 115 - No draft just yet
Migration and Data Portability

Motivations:

- Move existing user data between systems over generic API -> JMAP
- Give API spec to legacy systems which have no appropriate API
- Combine with other solutions to migration and portability-related problems

Spec should provide:

- Guidance for Simple API development
  - low requirements, reduced JMAP feature set
- Wild ideas Extensions for further migration-related problems
Essential Profile for rapid development

● “How to Quickstart JMAP”
  a. Bare minimum for one-time migration use
  b. Guidance on basics to start with

Rough outline of essential profile:

● Focus on key objects and properties for migration use-case
  ○ e.g. no /query, no properties in /get request
● Skip Session Object in case username == account
● Introduce simplified request
● No batching
Essential Profile for rapid development

Introduce simplified request (no need for JSON parsing on server):

$ curl <jmap-endpoint>?class=Preferences&method=get&accountId=<account-id>

$ curl <jmap-endpoint> --data '{
  "using": [
    "https://www.audriga.eu/jmap/preferences/",
    "https://www.audriga.eu/jmap/preferences-blocklist/",
    "urn:ietf:params:jmap:core"
  ],
  "methodCalls": [
    [
      "Preferences/get",
      {
        "accountId": "<account-id>"
      },
      "0"
    ]
  ]
}'

Note from IETF 114: Path like <jmap-endpoint>/<account-id>/<object>/<method>/<ids> would add routing requirement
Extension: JMAP Debug

https://www.audriga.eu/jmap/debug/

Our Problem:

- API software is running on a third-party infrastructure
- Log messages cannot be sent to an actual logging service
- Only a limited set of clients has access to a JMAP API

Solution: Send logs alongside response to a client

Other benefits:

- Use one software/channel instead of two!
- JMAP as a log protocol?
- ???

Limitations: Restrict access! (sensitive data)
Extension: JMAP Debug

```json
{
    "logs" : [
        {
            "file" : "Logger.php",
            "level" : "info",
            "line" : 32,
            "message" : "Array Logger has been successfully initialized",
            "timestamp" : "2022-01-18T10:26:56+01:00"
        },
        {
            "file" : "ErrorHandler.php",
            "level" : "warning",
            "line" : 52,
            "message" : "fopen(rcmail.php): failed to open stream: No such file or directory",
            "timestamp" : "2022-01-18T10:26:56+01:00"
        },
        {
            "file" : "ErrorHandler.php",
            "level" : "warning",
            "line" : 52,
            "message" : "fopen(rcmail_output_html.php): failed to open stream: No such file or directory",
            "timestamp" : "2022-01-18T10:26:56+01:00"
        },
        ...
    ],
    "methodResponses" : [
        {
            "Contact/get",
            ...
        }
    ]
}
```
Extension: Backend Info

Problem: No server software is perfect. How to let client handle those in an automated fashion?

Get Info about the backend software!

Without info you cannot act upon it.
Extension: Backend Info

```json
{
  "capabilities": {
    "urn:ietf:params:jmap:core:backendinfo": {
      "jmapBackend": "OpenXPort/Horde v1.0.0",
      "jmapProduct": "Horde Webmailer v1.0.0",
      "jmapEnvironment": "PHP v5.5"
    }
  },
  "accounts": {
    "A13824": {
      "name": "john@example.com",
      "accountCapabilities": {
        "urn:ietf:params:jmap:sieve:backendinfo": {
          "jmapProductExtension": "Horde Ingo v1.0.0",
          "jmapSieveScriptType": "SIEVE/HORDE",
          "jmapSieveProduct": "Cyrus timsieved"
        },
        ...
      }
    }
  }
}
```
Future work for Migration and Data Portability

- Create a draft