MyDataShare Developer Guide

This document describes the functionality currently available to ecosystem members.

This document is best approached with knowledge on MyData as a concept, MyDataShare as a solution and REST interfaces as well as OpenID Connect as technologies.

This document is a gentle introduction to utilizing MyDataShare functionalities.

It is complemented by deeper dives into the actual application programming interface as well as integration to the MyDataShare identity provider.

Disclaimer

MyDataShare is a work in progress.

Thus implementation details and processes are subject to change.

Do not hesitate to contact us in case you encounter bugs, inconsistencies or other hindrances.

Version

1.3 Updated 6.10.2021

Definitions and assumptions

This document utilizes the beta environment of MyDataShare.

This document utilizes Signicat as the identity provider.

This document utilizes curl as the command line tool to access the API. You can substitute any other tool that enables use of direct REST interactions (postman, insomnia). Remember that you can use the full MyDataShare API reference to use the Swagger UI to dig deeper into the endpoints.

The following definitions are used in the document for brevity, the URLs mentioned here are for the Beta environment, you will receive other URLS if you are supposed to use other environments:

- `AUTH_URL` = `https://gluu.beta.eks.mydatashare.com`
- `ORG_URL` = `https://api.beta.eks.mydatashare.com/organization/v2.0`
- `PUB_URL` = `https://api.beta.eks.mydatashare.com/public/v2.0`
- `WALLET_URL` = `https://wallet.beta.eks.mydatashare.com`
- `O_TOKEN` = `<access token as received from Gluu>`

Some included responses are abbreviated for brevity, as shown by ellipsis '...'.

The request-id field is an MyDataShare internal reference that is used to trace issues and assist in platform development.

MyDataShare API overview

The MyDataShare API is a RESTful interface.

Each entity is identified with an UUID, and this identifier is used in retrieving entities as well as describing relations between entities.

Each entity is returned as a whole in the response, and the response also contains the entities that are referred to in the returned.

This process is applied recursively, so the response may contain multiple entities.

Data model in a nutshell

The following entities are the high-level object types used in MyDataShare:

- Organization (ORG)
  - company (or equivalent) entity
  - able to create data providers and data consumers.
- Data consumer (DC)
  - sometimes referred as data using service
  - an organization-owned entity
  - able to create processing records (= consent requests).
- Data provider (DP)
  - sometimes referred as data providing service
  - an organization-owned entity
  - able to serve data as a source for data consumers.
- Access Gateway (AGW)
  - an organization-owned entity
  - a component in front of the data source that communicates with MyDataShare and introspects the needed permissions
- Data subject
  - a person
  - able to receive, view and respond to processing records.
  - Note: there is no single object type for a data subject. A data subject is uniquely identified with any of the identifiers (ssn, phone number, etc) in a keychain.
- Processing record (PR)
  - a permission request created by a data consumer to a data subject on specified data provider's contents.
  - this can be of different type responding to different bases of processing personal data defined by GDPR eg. consent, legal_oblication and contract.
  - Note: a data provider can be omitted in some cases (e.g. permission to store contact details)

Other relevant components

Identity provider

The component which is responsible for the authorization of organization clients and authentication of users. There are different authentication methods that can be used in different environments. These can be found from the public endpoint `GET $PUB_URL/auth_items`

Preliminary steps

Obtain client credentials

The client credentials are created by a MyDataShare-team member, please provide the following information in your request:

- Organization name in Finnish and English.
- Whether both data consumer and a data provider are needed, or just one of them.
- Token endpoint authentication method (`client_secret_basic` or `client_secret_post`)

In case you are creating a front end application that uses MyDataShare authentication, also add:

- URLs where browser can be redirected after successful authentication
- URLs where browser can be redirected after successful logout

- Display name of the application
- Contact email addresses and phone numbers

As a response you will receive one or many (depending on your use case) sets of

- Client identifier (= `ORG_CLIENTID` below)
- Client secret (= `ORG_PASSWORD` below)

Request creation of an Organization

The organizations are created by a MyDataShare-employee, please provide the following information in your request:

- Organization name in Finnish and English.
- Organization registration number (i.e. Y-Tunnus for finnish companies).
- Organization country.
- Client identifier(s) to be attached to the organization (as created in the preceding step)

As a response you will receive:

- UUID of the newly created organization (= `ORG_UUID` below).

Use Signicat test users

As the beta environment should never contain real user identities, we utilize the handful of test users as provided by the identity brokering finnish trust network members. Please refer to Signicat documentation and select your test subjects.

Note As there is only a limited number of the test users, you are likely to see unexpected processing records and other data in the Wallets belonging to them - these records have been created by other users of the environment.

Familiarize yourself with MyDataShare API

The MyDataShare API reference (Swagger) provides full details on the programming interface. This works against the MyDataShare Beta environment.

Obtain an access token

Perform the following request to the MyDataShare identity provider to obtain an access token with the client credentials flow (`client_secret_post`):

(split into multiple lines for readability)

```bash
curl -X POST
  -d "grant_type=client_credentials&client_id=$ORG_CLIENTID&
      client_secret=$ORG_PASSWORD&scope=organization"
  $AUTH_URL/oxauth/restv1/token
```

Extract the token from the response, and store it in `O_TOKEN` (organization-scopen token) variable for convenience.

Validate that you are able to access the API and your organization

Perform the following request to the MyDataShare interface to see the details of your organization:

(split into multiple lines for readability)

```bash
curl -X GET
  -H "authorization: Bearer $O_TOKEN"
  -H "Content-Type:application/json"
  $ORG_URL/organization/$ORG_UUID
```

The response describes the organization itself and its identifiers as well as its client identifiers:

```JavaScript
{
    "organizations": {
        "726720a1-99d7-41f0-9b5f-4c0a085f50f6": {
            "default_language": "fin",
            "url_group_id": 11,
            "country": "FIN",
            "description": "The greatest of the great old ones...",
            "legal_entity_type": "oy",
            "translation_id": 25,
            "name": "Oy Cthulhu Ab",
            "suppressed_fields": [
                "data_consumers.uuid",
                "data_providers.uuid"
            ],
            "updated": "2020-06-24T10:34:09.982462+00:00",
            "organization_client_ids.uuid": [
                "31582023-5619-4dc2-a469-e1e9466fdd79"
            ],
            "uuid": "726720a1-99d7-41f0-9b5f-4c0a085f50f6",
            "deleted": false,
            "organization_ids.uuid": [
                "c5a0ae3f-25f7-4e1a-b693-34493c37beb0"
            ],
            "created": "2020-06-24T10:34:09.982457+00:00",
            "group_id": null
        }
    },
    "organization_ids": {
        "c5a0ae3f-25f7-4e1a-b693-34493c37beb0": {
            "org_id_value": "3385750-2",
            "uuid": "c5a0ae3f-25f7-4e1a-b693-34493c37beb0",
            "suppressed_fields": [
                "organizations.uuid"
            ],
            "deleted": false,
            "org_id_type": "registration_number",
            "created": "2020-06-24T10:33:05.209426+00:00",
            "updated": "2020-06-24T10:33:05.209431+00:00",
            "org_id_country": "FIN"
        }
    },
    "organization_client_ids": {
        "31582023-5619-4dc2-a469-e1e9466fdd79": {
            "name": "Client of Cthulhu",
            "client_id": "@!FF9F.35A9.9F4B.DF31!0001!6A12.9F11",
            "created": "2020-06-24T10:37:15.417749+00:00",
            "updated": "2020-06-24T10:37:15.417754+00:00",
            "uuid": "31582023-5619-4dc2-a469-e1e9466fdd79",
            "deleted": false,
            "organization_uuid": "726720a1-99d7-41f0-9b5f-4c0a085f50f6"
        }
    }
}
```

The identifiers correspond to real-world identifiers (such as the finnish y-code business registration number), and the client identifiers are used to authorize access to the interface (as described in the preceding step).

Each list of entities identifies them by their unique identifiers (UUIDs), which are used to refer to them and to link them to each other.

```JavaScript
"organizations": {
        "726720a1-99d7-41f0-9b5f-4c0a085f50f6": {
            "default_language": "fin",
            "uuid": "726720a1-99d7-41f0-9b5f-4c0a085f50f6",
            ...
```

Create a data provider

Perform the following request to the MyDataShare interface to create a data provider:

(split into multiple lines for readability)

```bash
curl -X POST -H "authorization: Bearer $O_TOKEN" -H "Content-Type:application/json"
    -d '{
        "organization_uuid": $ORG_UUID,
        "default_language": "eng",
        "name": "Name of the data provide",
        "description": "Description of the data provider",
        "has_live_preview": false,
        "static_preview": "This data contains the data x, y and z"
    }' $ORG_URL/data_provider
```

Parameter	Mandatory	Description	Example value
`organization_uuid`	Yes	Identifier of the organization that owns the data provider	3fa85f64-5717-4562-b3fc-2c963f66afa6
`access_gateway_uuid`	No	The uuid of a Access Gateway if there is one in front of the data source. See Create an Access Gateway-section for more details	e0bfa0ae-27b9-42b1-a90f-bfaa2e4a27dd
`name`	Yes	Name of the data provider	Delivery address
`description`	Yes	Description of the data provider	The information provided includes the following: street address, city, zip code, state and country
`has_live_preview`	Yes	Whether data subject specific data shall be displayed in the Wallet (not supported currently)	false
`static_preview`	Yes	A preview that is displayed in every data subject's Wallet	Malone Avenue 32, Stockton, 85010, UT, United States
`default_language`	Yes	The language in which the contents are presented by default	fin
`input_pr_identifier`	No	Should the introspection response include the identifier that is used in the PR request (eg. pairwise, or SSN)?	true
`input_id_types.uuid`	No	The uuid's of those id_types that should be included in the introspection response (eg. pairwise, email or SSN). These are needed if the AGW needs a different identifier attached to the data subject (for example to fetch the data from data source) that is not the one that is attached to the PR. The id_types can be fetched from endpoint `$ORG_URL/id_types`	["a0aee0bf-27b9-42b1-a90f-bfaa2e4a27dd", "726720a1-99d7-41f0-9b5f-4c0a085f50f6"]

The response describes the created data provider:

```JavaScript
    "data_providers": {
        "a0aee0bf-27b9-42b1-a90f-bfaa2e4a27dd": {
            "url_group_id": 1474,
            "translation_id": 228,
            "suppressed_fields": [
                "identifiers.uuid",
                "processing_records.uuid"
            ],
            "default_language": "fin",
            "has_live_preview": false,
            "name": "Nimi",
            "static_preview": "Esikatselu",
            "created": "2020-06-18T09:08:50.462913+00:00",
            "updated": "2020-06-18T09:08:50.462918+00:00",
            "major_version": 1,
            "organization_uuid": "726720a1-99d7-41f0-9b5f-4c0a085f50f6",
            "minor_version": 0,
            "uuid": "a0aee0bf-27b9-42b1-a90f-bfaa2e4a27dd",
            "description": "Kuvaus",
            "deleted": false
        }
    ...
```

Take note of the uuid identifier of the data provider (= `DP_UUID`). In this example: `a0aee0bf-27b9-42b1-a90f-bfaa2e4a27dd`.

Create a data consumer

As the data consumer contains the purpose, description and legal details that are copied to the processing record, data consumer can be thought of as a permission template from which the PR derives its content from. Processing record is what the data subject will see in the MyDataShare Wallet or service UI.

Perform the following request to the MyDataShare interface to create a data consumer:

(split into multiple lines for readability)

```Bash
curl -X POST -H "authorization: Bearer $O_TOKEN" -H "Content-Type:application/json"
    -d '{
        "organization_uuid": $ORG_UUID,
         "default_language":"eng",
         "name":"Nimi",
         "purpose":"Your data x will be used to provide you service y",
         "description":"The organization z is requesting your permission to do abc...",
         "legal":"The more detailed legal implications and data usage descriptions are...",
         "pre_cancellation":"If you revoke this consent, your data x is deleted.",
         "post_cancellation":"As you revoked the consent, your data x is going to be deleted."
    }' $ORG_URL/data_consumer
```

Parameter	Mandatory	Description	Example value
`organization_uuid`	Yes	Identifier of the organization that owns the data consumer	3fa85f64-5717-4562-b3fc-2c963f66afa6
`name`	Yes	Name of the data consumer	Delivery details
`description`	Yes	Description of the data consumer	In order to be able to ensure a smooth delivery, the recipient is requested to provide all pertinent details. These details include ...
`purpose`	Yes	Purpose of the processing records derived from this data consumer	To use the delivery address to deliver the products to the correct address
`legal`	Yes	Legal flavoured purpose of the the processing records derived from this data consumer. This is shown in the UI if the user clicks this open, thus this can be longer text.	The delivery details provided will be forwarded to all agents involved in the delivery process. The list of commonly utilized agents is available at ... (this usually includes needed legal details eg. a more detailed description of the data and links to relevant policies and terms pages or documents.
`pre_cancellation`	Yes (can be empty)	Text displayed to the data subject at permission declination/revocation	If you revoke access to your delivery details, the delivery of the goods may be significantly delayed.
`post_cancellation`	Yes (can be empty)	Text displayed to data subject after permission declination/revocation	You have now revoked access to your delivery details. Any delays experienced during the delivery process have been explicitly disclaimed by us.
`default_language`	Yes	The language in which the contents are presented by default	fin

The response describes the created data consumer:

```JavaScript
{
    "data_consumers": {
        "95322354-b116-47e3-b210-d3d672f8139f": {
            "url_group_id": 1472,
            "translation_id": 226,
            "legal": "Laki.",
            "post_cancellation": "Post",
            "default_language": "fin",
            "name": "Nimi",
            "suppressed_fields": [
                "processing_records.uuid"
            ],
            "created": "2020-06-18T09:03:56.760426+00:00",
            "updated": "2020-06-18T09:03:56.760431+00:00",
            "major_version": 1,
            "purpose": "Tarkoitus",
            "organization_uuid": "726720a1-99d7-41f0-9b5f-4c0a085f50f6",
            "minor_version": 0,
            "uuid": "95322354-b116-47e3-b210-d3d672f8139f",
            "pre_cancellation": "Pre",
            "description": "Kuvaus",
            "deleted": false
        }
    ...
```

Take note of the uuid identifier of the data provider (= `DC_UUID`). `95322354-b116-47e3-b210-d3d672f8139f` in this example.

Create a processing record

Perform the following request to the MyDataShare interface to create a processing record that defines a consent request for a data subject to access content at a data provider:

(split into multiple lines for readability)

```bash
curl -X POST -H "authorization: Bearer $O_TOKEN"
    -H "Content-Type:application/json"
    -d '{
        "data_consumer_uuid":"$DC_UUID",
        "data_provider_uuid":"$DP_UUID",
        "identifier": {
            "country":"FIN",
            "type":"ssn",
            "id":"TESTUSER-SSN"
        },
        "record_type":"consent"
    }'
    $ORG_URL/processing_record
```

Parameter	Mandatory	Description	Example value
`data_provider_uuid`	Yes	Identifier of the data consumer	3fa85f64-5717-4562-b3fc-2c963f66afa6
`data_consumer_uuid`	Yes	Identifier of the data provider	3fa85f64-5717-4562-b3fc-2c963f66afa6
`record_type`	Yes	Legal basis for permission request	consent
`identifier.type`	Yes	Type of the data subject's identifier	ssn
`identifier.country`	Yes	Country of the data subject's identifier	fin
`identifier.id`	Yes	Value of the data subject's identifier	220188-145U

The response describes the created processsing record (and all related entities):

```javascript
    "processing_records": {
        "7b5b7c9d-a9fb-41c2-9fd7-673f3a23b61b": {
            "data_provider_uuid": "a0aee0bf-27b9-42b1-a90f-bfaa2e4a27dd",
            "record_type": "consent",
            "legal": null,
            "status": "pending",
            "name": null,
            "supersedes_uuid": null,
            "expires": null,
            "accepted_language": null,
            "identifier_uuid": "7bf3dc27-d69b-4225-a100-0da9710cbc48",
            "not_valid_before": null,
            "url_group_id": 1475,
            "description": null,
            "created": "2020-06-18T09:10:20.268744+00:00",
            "updated": "2020-06-18T09:10:20.268749+00:00",
            "purpose": null,
            "data_consumer_uuid": "95322354-b116-47e3-b210-d3d672f8139f",
            "uuid": "7b5b7c9d-a9fb-41c2-9fd7-673f3a23b61b",
            "group_id": null,
            "deleted": false
        }
    },
```

Take note of the identifier of the processing record (= `PR_UUID`).

The most interesting returned entity, in addition to the processing record itself, is the identifier and its relatives that describes the data subject, the person to whom this processing record is addressed to:

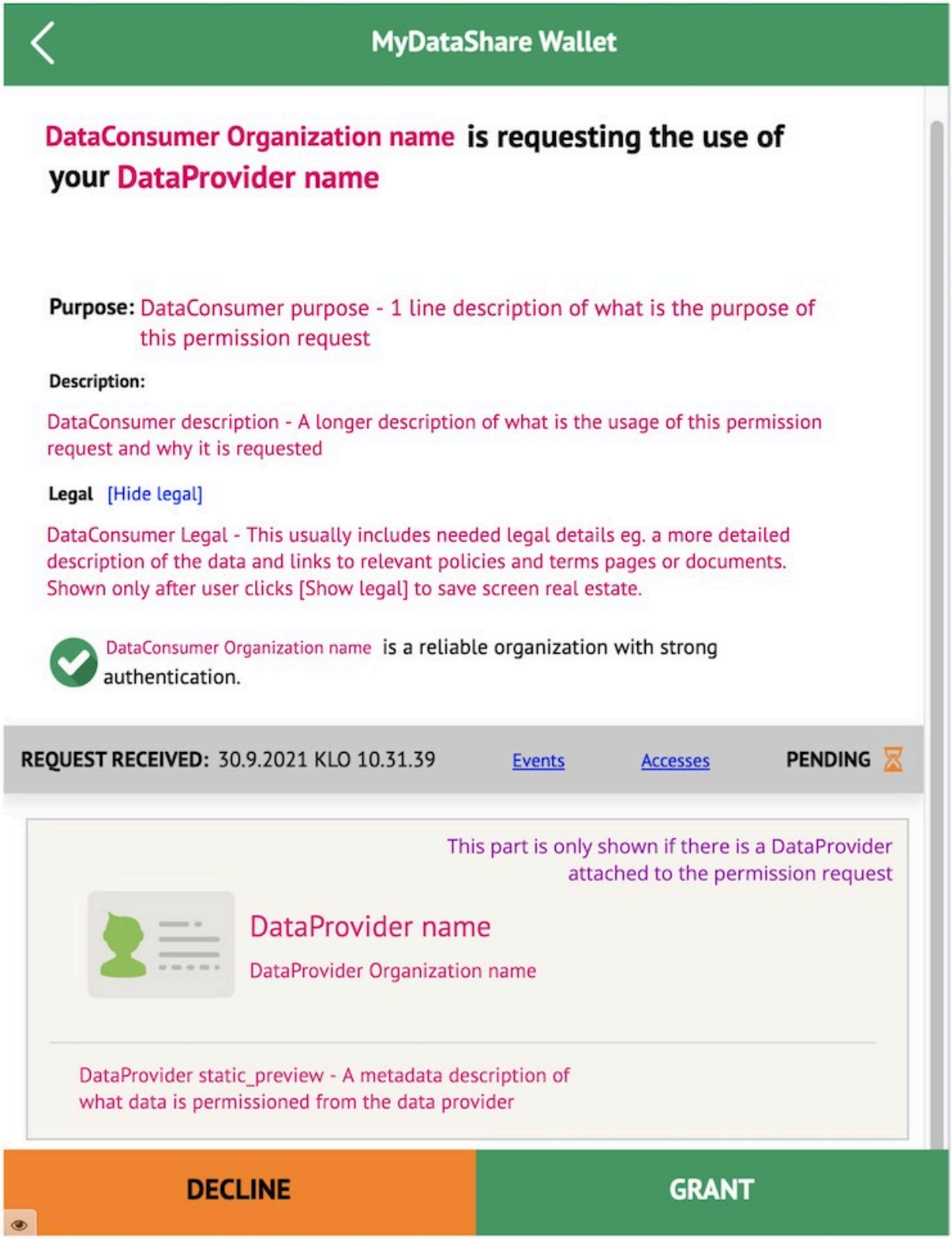
```javascript
    "identifiers": {
        "7bf3dc27-d69b-4225-a100-0da9710cbc48": {
            "url_group_id": 311,
            "verified": null,
            "suppressed_fields": [
                "data_providers.uuid",
                "processing_records.uuid"
            ],
            "id_type_uuid": "49679532-9c4c-4a46-9e5b-a914472f9612",
```

```
            "created": "2020-06-12T10:39:11.567107+00:00",
            "updated": "2020-06-12T10:39:11.567108+00:00",
            "id": "010200A9618",
            "id_provider_infos.uuid": [
                "9cf5cfe3-9412-4a31-b4f8-dabb191d260e",
                "d199be2f-d8f4-4a9f-a778-c7f69fe75d87"
            ],
            "uuid": "7bf3dc27-d69b-4225-a100-0da9710cbc48",
            "group_id": null,
            "deleted": false
        }
    },
    ...
    "id_providers": {
        "4e161a9e-f1f3-4e89-b30a-6854730bcda4": {
            "url_group_id": 1463,
            "translation_id": 222,
            "type": "gluu",
            "suppressed_fields": [
                "identifiers.uuid"
            ],
            "name": "Gluu",
            "created": "2020-06-12T11:07:07.850535+00:00",
            "updated": "2020-06-12T11:07:07.850540+00:00",
            "id": "https://gluu.beta.eks.mydatashare.com",
            "uuid": "4e161a9e-f1f3-4e89-b30a-6854730bcda4",
            "description": "Gluu (beta)",
            "deleted": false
        }
    },
    ...
    "id_provider_infos": {
        "9cf5cfe3-9412-4a31-b4f8-dabb191d260e": {
            "first_name": "Onni Juhani",
            "language": null,
            "last_name": "Korhonen",
            "created": "2020-06-18T08:56:21.056922+00:00",
            "updated": "2020-06-18T08:56:21.056929+00:00",
            "last_login": "2020-06-18T08:56:20.959805+00:00",
            "attributes": null,
            "source": "signicat",
            "uuid": "9cf5cfe3-9412-4a31-b4f8-dabb191d260e",
            "identifier_uuid": "7bf3dc27-d69b-4225-a100-0da9710cbc48",
            "id_provider_uuid": "4e161a9e-f1f3-4e89-b30a-6854730bcda4",
            "deleted": false
        },
      },
      ...
    "id_types": {
        "49679532-9c4c-4a46-9e5b-a914472f9612": {
            "url_group_id": 1,
            "translation_id": 1,
            "type": "ssn",
            "suppressed_fields": [
                "identifiers.uuid"
            ],
            "name": "Finnish personal identity number",
            "created": "2020-06-12T10:37:07.518073+00:00",
            "updated": "2020-06-12T10:37:07.518073+00:00",
            "country": "FIN",
            "uuid": "49679532-9c4c-4a46-9e5b-a914472f9612",
            "verify_interval": -1,
            "description": "Identifier for tracking individuals",
            "deleted": false
        }
    }
```


The MyDataShare Wallet

MyDataShare Wallet is the interface a data subject can use to browse and react to the permission requests. The data subject can decline or grant `consent` type permission requests and see the status of permission requests that eg. of type `legal_obligation` and `contract`.

In MyDataShare beta environment the wallet can be found from the URL https://wallet.beta.eks.mydatashare.com/front In this document this is refereced in examples with the variable `WALLET_URL`



This screenshot has different text values indicated in purple to show how they are shown to the user and to know where their content comes from.

Create an access gateway (AGW)

Access gateway is a component in front of the data source that communicates with MyDataShare and introspects the needed permissions. MyDataShare has a reference implementation of a Python code that can be used to implement this functionality. It can be found here Access Gateway GitHub repo. More information about it in its own README.

> Access Gateway (hereafter, for brevity: AGW) is a general purpose extendable API gateway developed originally for MyDataShare MyData operator platform to shield and protect the data providing services but is not limited to be used within the MyDataShare ecosystem. Vastuu Group wants through open sourcing the code as a baseline/reference implementation of the MIM4 Connector function to invite vendors and developers to build and extend on it and simultanously to create multi-vendor interoperability between MyData operators with heterogenous product realizations. However, most of this document focuses on using MyDataShare Access Gateway as part of MyDataShare ('MDS') ecosystem. The MIM4 Connector specific configuration and use of AGW will be documented in detail as the MIM4 project evolves forward. The latest on MIM4 Connectivity Specification and the Connector can be found here.
>
> The main feature of AGW is to help customers easily protect their data provider (MyData term: data source) services. AGW acts as a shield atop the data provider service and introspects inbound requests from data consumers (MyData term: data using service) for validity.

Perform the following request to the MyDataShare interface to create an access gateway:

(split into multiple lines for readability)

```bash
curl -X POST -H "authorization: Bearer $O_TOKEN" -H "Content-Type:application/json"
    -d '{
        "organization_uuid": "1fade972-f8d8-49aa-b3b8-111111111111",
        "name": "Access gateway in front of data source X",
        "description": "The access gateway in front of the data source X ..."
     }' $ORG_URL/access_gateway
```

Parameter	Mandatory	Description	Example value
`organization_uuid`	Yes	Identifier of the organization that owns the access gateway	3fa85f64-5717-4562-b3fc-2c963f66afa6
`name`	Yes	Name of the access gateway	Access gateway in front of data source X
`description`	Yes	Description of the access gateway	The access gateway in front of the data source X validating that the access to this data should be permitted

The response describes the created access gateway:

```javascript
    "access_gateways": {
        "e0bfa0ae-27b9-42b1-a90f-bfaa2e4a27dd": {
          "suppressed_fields": [
            "client_ids.uuid",
            "data_providers.uuid"
          ],
          "url_group_id": 6975,
          "uuid": "e0bfa0ae-27b9-42b1-a90f-bfaa2e4a27dd",
          "organization_uuid": "e9721fad-f8d8-49aa-b3b8-9810b39e8ba3",
          "description": "The access gateway in front of the data source X validating...",
          "updated": "2021-10-01T12:33:37.025606+00:00",
          "name": "Access gateway in front of data source X",
          "created": "2021-10-01T12:33:37.025603+00:00",
          "deleted": false
        }
    ...
```

Take note of the identifier of the access gateway (= AGW_UUID). In this example: `e0bfa0ae-27b9-42b1-a90f-bfaa2e4a27dd`.

Add the access gateway URL

(split in to multiple lines for readability)

```bash
curl -X POST -H "authorization: Bearer $O_TOKEN" -H "Content-Type:application/json"
  -d '{
    "url": "https://api.mydatashare.com/introspect",
    "url_type": "access_gateway",
    "method_type": "post",
    "name": "AGW url for data source x",
    "description": "AGW url for data source x in environment y for the purpose of z..",
  }'
  $ORG_URL/url
```

Response

```javascript
{
  "uuid": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "created": "2021-10-05T19:07:21.321Z",
  "updated": "2021-10-05T19:07:21.321Z",
  "deleted": false,
  "suppressed_fields": [
    [
      "data_providers.uuid",
      "processing_records.uuid"
    ]
  ],
  "url_uuid": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "url_group_id": 0,
  "subgroup_id": 0
}
```

Save the uuid indentifier (= URL_UUID).

Add the URL to the Access Gateway

The Access Gateway URL needs to be attached to the Access Gateway so that the JWT validation of the request ticket will pass. The Access Gateway URL acts as the audience for the JWT.

```bash
curl -X POST -H "authorization: Bearer $O_TOKEN" -H "Content-Type:application/json"
  -d '{
    "url_uuid": "$URL_UUID",
    "model": "access_gateway",
    "model_uuid": "$AGW_UUID"
}'
  $ORG_URL/url_group_url
```

Add the URL to the Data provider

```bash
curl -X POST -H "authorization: Bearer $O_TOKEN" -H "Content-Type:application/json"
  -d '{
    "url_uuid": "$URL_UUID",
    "model": "data_provider",
    "model_uuid": "$DP_UUID"
}'
  $ORG_URL/url_group_url
```

Data retrieval through the Access Gateway

Please see the public The Access Gateway repo for up-to-date instructions on how the data can be retrived through the Access Gateway.

View the processing record in the MyDataShare Wallet

Log in to the MyDataShare Wallet at: https://wallet.beta.eks.mydatashare.com/ (use the same test user to whom you targeted with the processing record in the preceding step).

You can interact with the processing record in the Wallet (grant the request, decline the request, re-grant the request).

Obtain status of the processing record

Perform the following request to the MyDataShare interface to retrieve the processing record:

(split in to multiple lines for readability)

```bash
curl -X GET
  -H "authorization: Bearer $TOKEN"
  -H "Content-Type:application/json"
  $ORG_URL/processing_record/$PR_UUID
```

The `status`-field describes the current status of the processing record.

```javascript
...
"status":"active"
...
```

Add translation to a data provider or a data consumer

MyDataShare supports localized text fields for many object types.

Using an access token with organization-domain scope, it is possible to add translations to data providers and data consumers.

(split into multiple lines for readability)

```bash
curl -X POST
    -H "authorization: Bearer $O_TOKEN"
    -H "Content-Type:application/json"
    -d '{
        "object_type": "data_provider",
        "object_uuid": "$DP_UUID",
        "translations": [
            {
              "language": "swe",
              "field_name": "name",
              "translation": "översuttit namn"
            },
            {
              "language": "swe",
              "field_name": "description",
              "translation": "översuttit berskrivining"
            }
        ]
    }'
    $ORG_URL/translation
```

You must add all the translations of non-optional translatable fields with a single request. Failure to submit all of them is signposted as follows:

```javascript
{
  "error": "bad_request",
  "description": "Missing required translation fields (['description'])",
  "request_id": "REQ-ID"
}
```

Create an application that uses MyDataShare

The interfaces as described in the preceding chapter are usable programmatically as well. The requests and responses are plain REST. Generating and interpreting them is significantly easier using a convenience library (obviously dependent on the prorgramming language used).

If you implement your solution as a web application, it is strongly suggested to split the application into two parts, and confine the use of any client secret in the backend component only.

However, client applications (native mobile applications or Javascript based single-page applications) that cannot keep the client secret secure can use the Authorization Code Grant Flow with PKCE extension.

Please refer to the accompanying MyDataShare OpenID Connect Integration Specification for details about the authentication flows of the application and integration in general.

Allow the user to interact with the MyDataShare Wallet

You can forward the user's browser to a single processing record (and return back to your application) using the following URL:

(split in to multiple lines for readability)

```bash
$WALLET_URL/wallet/requests?
  processing_record=xxx&
  return_url=https://go.back.here.com&
  auth_item_uuid=yyy#pr
```

Where auth*item*uuid is an optional reference to a selected authentication method, if this is known. This will direct the user to this authentication flow, if the session does not already exist. The different authentication items and their uuids can be fetched from this URL: `$PUB_URL/auth_items`.

Optionally you can add the string `lng=en` (before the `#pr` part) to force the language to English. Supported languages are fi (Finnish), en (English) and sv (Swedish).

API reference

The MyDataShare API reference provides full details on the programming interface.