# d.velop documents repo export API

# Table of Contents

# 1. d.velop documents repo export API

## 1.1. Introduction

This section provides you with general product information.

### 1.1.1. About the d.velop documents repo export API

The d.velop documents repo export API (export API) is a REST HTTP endpoint through which you can retrieve metadata, links to files and system configurations (master data). Starting from the migration interface, the export API assists with the export of documents from d.velop documents.

The output format for the metadata information is StandardDocumentImportFormat (JSON). This format is the defined target format for the migration API and contains practically all the metadata information for a d.3 document. Thus, this interface enables the general export of documents from d.velop documents. The document export is provided in both the cloud and on-premises (in the currently supported versions). This makes it easy to perform direct migrations from the d.velop on-premises platform to the cloud platform. It highlights potential solutions for customers to exit the cloud platform of d.velop cloud again (exit strategy) or to perform backups themselves.

The export API is available in the cloud via a dmsdocs app endpoint. In on-premises systems, you can install the export API via the d.3 server tool setup. Furthermore, you can export original files directly from storage systems with the export API as of d.ecs storage manager version 3.3.0. The current API development version does not support encrypted directory structures (document trees). Therefore, original files cannot be exported from these systems.

You can operate the export API either via HTTP or through a client solution via d.velop documents repo exporter.

### 1.1.2. Scope of functions of d.velop documents repo export API

The export API includes the following functions and key data:

- HTTP-based REST endpoint
- Available as a separate setup without updating the system
- Available through all platforms
- Original files can also be exported from storage systems
- Output format is already in the appropriate import format for cloud migrations
- Delta export capabilities
- Export can be restricted using filters (time, category, document IDs)
- Stateless
- Logging in d.velop logfile viewer (d.3 logview)

### 1.1.3. Prerequisites

As of the version Current 2023.Q4, the d.velop documents repo export API in the d.3 server package (and therefore in all currently supported feeds) is an integral part of the d.velop documents component dmsdocs. You can activate the application here without installing additional software. For older versions of d.velop documents, you can install the d.velop documents repo export API with the d.3 server tools package using the d.velop software manager.

## 1.2. Using the API functions

Below you can learn about the various functions on the user interface of the d.velop documents repo export API (export API).

### 1.2.1. Activating the API for the cloud

If you want to use the API for the cloud, you first have to activate the API for each d.velop tenant.

**This is how it works**

1. Contact d.velop support. As the heading, use “Activation of the export interface for multi-tenants”.
2. Set up a user with export permissions. Use the following API to do so: Configuring user rights for the cloud

You can access the API under:

```
https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/export
```

### 1.2.2. Activating the API for on-premises

You can install the export API (on-premises) in existing d.velop documents systems at a later point in time using the tool setup. This standalone application provides a REST-capable web server that makes the export interface and the file download (FileDownload) interface available.

> **Note**
> As of d.3 server version Current 2023.Q4, the export API can also be accessed via **dmsdocs.exe** as standard. A separate installation is thus not required.
>
> In this case, you enter the parameter **ENABLE_EXPORTS = "1"** in the file **addon.ini**. You also configure a user with export permissions.

**This is how it works**

1. Start the application via Windows command prompt (**cmd**) as follows:

   ```
   d.velop_documents_repo_export_API.exe -c <drive name>:\<path to
   d3config> -u <exportUser> -p <Port (Default: 8333)>
   ```

   You can access the export API under the following URL:

   ```
   http://<local IP>:<Port>/repoexport/export
   ```
2. Ensure that the user you are using exists and, depending on the version, has administrative permissions or permissions for exporting data.
3. Use this user for every request to the export API with basic authentication. Alternatively, you can use the API key of an authorized user if you are using the dmsdocs app for the export.

### 1.2.3. Defining the export filter

You may have to add the base URL to URLs that have been returned by the API.

> **Warning**
> Security information: Note that authentication information is permitted to be sent only to the configured API endpoint.

The export process consists of multiple steps:

1. An export job is created that describes the filter and the number of parallel export processes.
2. A further query is performed for each returned batch URL in the “batches” list. For more information, see: Requirements for metadata and download URLs
3. After you call a batch URL, the result includes a determination of whether there is another page for the batch. Any “next” link present must be followed until the returned resource does not contain any further entry for “next”.

4. Once all the parallel requests are complete, the export ends.

**Request**:

Initial request for an export using a filter to limit the documents provided for the export.

- dmsdocs app: **PUT https://<base URL>/dmsdocs/r/<repository ID>/export**
- Standalone application: **PUT http://<local IP>:8333/repoexport/export**
- Content type: **application/json**
- Accept: **application/json**

```
{
    "modifiedAfter": "2017-11-07T13:27:37.643Z",
    "modifiedBefore": "2021-11-07T13:27:37.643Z",
    "numberOfProcesses": 4,
    "documentTypesByD3Id": [
        "1e4b6",
        "5fa04"
    ],
    "documentTypesById": [],
    "docIds": [],
    "batchSize": 10,
    "migrated" : false
    "useSynchIdFormat": false
}
```

| Tag name | Description |
|---|---|
| **documentTypeById** | Category to which the filter is applied. Use this tag for GUIDs. If you do not define a category, all the documents are selected. |
| **documentTypeByD3Id** | Category to which the filter is applied. Use this tag for d.3 IDs. If you do not define a category, all the documents are selected. |
| **modifiedAfter** | Timestamp in accordance with ISO 8601. This timestamp is used for delta exports and marks the last time that a document was processed (**overAllProcDate**). Documents that were processed at exactly the time of the timestamp are not filtered out. The timestamp must be compatible with the underlying DBMS. |
| **modifiedBefore** | Timestamp in accordance with ISO 8601. This timestamp is used for delta exports and marks the last time that a document was processed (**overAllProcDate**). Documents that were processed at exactly the time of the timestamp are not filtered out. The timestamp must be compatible with the underlying DBMS. |
| **docIds** | Special list of document IDs (**docIds**) to be exported. The server can currently process a maximum of 100 document IDs. |
| **batchSize** | Maximum block size (number of documents) that can be retrieved via a GET request. |
| **numberOfProcesses** | The number of processes planned to access the export API in parallel. |
| **migrated** | Limits the filter to documents created as a result of migration. |
| **useSynchIdFormat** | Export with a unique property ID for metadata. |

**Response**:

Content type: **application/json**

```
{
    "documentsToExportCount": "164",
    "filter": {
        "modifiedAfter": "2017-11-07T13:27:37.643Z",
```

```
        "modifiedBefore": "2023-01-01T01:01:01.001Z",
        "numberOfProcesses": 4,
        "documentTypesByD3Id": [],
        "documentTypesById": [],
        "docIds": [],
        "batchSize": 200
    },
    "batches": [
        "/dmsdocs/r/a88362cb-f626-5fc2-9293-61f6eafc4b0b/export?
startDocId=DE00000000&batchNumber=0&mB=2021-11-07T13:27:37.643Z&mA=2017-11-0
7T13:27:37.643Z&dTD3Id=1e4b6,5fa04&bS=10&nP=4&",
        "/dmsdocs/r/a88362cb-f626-5fc2-9293-61f6eafc4b0b/export?
startDocId=G200000001&batchNumber=1&mB=2021-11-07T13:27:37.643Z&mA=2017-11-0
7T13:27:37.643Z&dTD3Id=1e4b6,5fa04&bS=10&nP=4&",
        "/dmsdocs/r/a88362cb-f626-5fc2-9293-61f6eafc4b0b/export?
startDocId=M100000007&batchNumber=2&mB=2021-11-07T13:27:37.643Z&mA=2017-11-0
7T13:27:37.643Z&dTD3Id=1e4b6,5fa04&bS=10&nP=4&",
        "/dmsdocs/r/a88362cb-f626-5fc2-9293-61f6eafc4b0b/export?
startDocId=M200000007&batchNumber=3&mB=2021-11-07T13:27:37.643Z&mA=2017-11-0
7T13:27:37.643Z&dTD3Id=1e4b6,5fa04&bS=10&nP=4&"
    ]
}
```


| Tag name | Description |
|---|---|
| **documentsToExportCount** | The number of documents that meet the filter criteria. |
| **filter** | Confirmation of the included filter criteria: |
| **batches** | Links to export batches (specified relatively to the export API). The metadata and download URL are provided via GET request under these addresses. In this case, the link contains the batch number and the associated filter criteria. |

### 1.2.4. Requirements for metadata and download URLs

In the example below, you can learn how to use the batch URLs generated in the section “Defining the export filter” to request the relevant metadata for the filter and the download URLs of the associated usage files.

You obtain the URLs for retrieving the metadata by creating an export filter or by following the “next” links in the metadata.

> **Note**
> For a complete export, you have to retrieve all the pages from all the batches provided.

**Request**:

- dmsdocs app: **GET https://<base URL>/<URL from batch or “next” link>**
- Standalone application: **GET http://<local IP>/<URL from batch or “next” link>**
- Accept: **application/json**

**Response**:

Sample structure of a response (content type: **application/json**):


```
{
    "docs": [
        {
            "files": [
```

```
                {
                    "fileId": 1,
                    "filename": "DE00000000.1",
                    "downloadUrl": "/dmsdocs/r/a88362cb-
f626-5fc2-9293-61f6eafc4b0b/Document/DE00000000/files/1/root"
                },
                {
                    "fileId": 1,
                    "downloadUrl": "/dmsdocs/r/a88362cb-
f626-5fc2-9293-61f6eafc4b0b/Document/DE00000000/files/1/P1",
                    "dependentExtension": "P1"
                }
            ],
              "metadata": {
                  "docId": "DE00000000",
                  [...]
            }
        }
    ],
        "errorDocs" : [
                {
                        "docId": "DE00000001",
                        "message": "<Beschreibung eines Exportfehlers>"
                }
        ],
    "_links": {
        "next": {
            "href": "/dmsdocs/r/a88362cb-f626-5fc2-9293-61f6eafc4b0b/export?
startDocId=DE00000004&batchNumber=4&mB=2021-11-07T13:27:37.643Z&mA=2017-11-0
7T13:27:37.643Z&dTD3Id=1e4b6,5fa04&bS=1&nP=4&"
        },
        "self": {
            "href": "/dmsdocs/r/a88362cb-f626-5fc2-9293-61f6eafc4b0b/export?
startDocId=DE00000000&batchNumber=0&mB=2021-11-07T13:27:37.643Z&mA=2017-11-0
7T13:27:37.643Z&dTD3Id=1e4b6,5fa04&bS=1&nP=4&"
        }
    }
}
```


**Description of the tag**:

| Tag name | Description |
|---|---|
| **_links.self** | Link to your export batch. |
| **_links.next** | Link to the next batch that is used to provide additional documents and metadata. If no link is present, this batch has been fully queried. |
| **docs** | Item containing the metadata and information for the files. |
| **errorDocs** | List of errors that occurred while processing the export batch. Documents in this list could not be fully exported. |
| **files** | Combines all the information for the files into a metadata record. |
| **metadata** | Document information in the standard import format (importable format). |

## 1.2.5. Exporting the source system configuration

**Request: Querying master data properties**

- dmsdocs app: **GET https://<base URL>/dmsdocs/r/<repository ID>/objectmanagement/properties**
- Standalone application: **GET http://<local IP>:8333/repoexport/objectmanagement/properties**

- Content type: **application/json**

Response:

```
{
  "_embedded": {
    "properties": [
      {
        "dataType": "CHAR",
        "id": 6,
        "isMultiValue": false,
        "name": {
          "de": "Alphanumerisch"
        }
      },
      {
        "dataType": "CHAR",
        "id": 7,
        "isMultiValue": false,
        "name": {
          "de": "Alphanumerisch2"
        }
       },...
        ]
    }
}
```

**Request: Querying master data categories**

- dmsdocs app: **GET https://<base URL>/dmsdocs/r/<repository ID>/objectmanagement/categories**
- Standalone application: **GET http://<local IP>:8333/repoexport/objectmanagement/categories**
- Content type: **application/json**

Response:

```
{
  "_embedded": {
    "categories": [
      {
        "id": "AAKTE",
        "name": {
          "de": "Akte"
        },
        "propertyRefs": [
          {
            "propertyId": "DOCUMENT_ID"
          },
          {
            "propertyId": "CATEGORY"
           },...
                ]
           }
         ]
    }
}
```

### 1.2.6. Configuring user rights for the cloud

You can use the HTTP REST endpoint **user** from **dmsdocs.exe** to display existing users that are known to the d.3 server. You can grant export and migration permissions to these users. This HTTP endpoint can be called only by users with administrative rights.

You can restrict user lists using query parameters. You can also display and edit individual users.

### Structure of an individual user object

Main route:

**https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user**

Link to the individual user object:

```
{
  "_links": {
     "self": {
         "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/d3Id/user1",
         "templated": false
         }
   },
```

d.3 ID of the user:

```
"d3Id": "user1",
```

IDP ID of the user:

```
"idpId": "7B841E93-EC4E-4790-B9D7-AD7F5DFCC82B"
```

Define whether you want the user to have export permissions:

```
"hasExportRight": true,
```

Define whether you want the user to have migration permissions:

```
"hasMigrationRight": true,

}
```

When you change a user permission, call the self link of the user object using a PUT request. Send the received information to the user as the JSON body in the request as well.

### GET requests and query parameters

**Objective: Querying all users**

Request/subroute: **GET https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user**

Sample end point response:

```
{
    "user": [
        {
            "_links": {
                "self": {
                    "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/
d3Id/user1",
                    "templated": false
```

```
                }
            },
            "d3Id": "user1",
            "hasExportRight": true,
            "hasMigrationRight": true,
            "idpId": "7B841E93-EC4E-4790-B9D7-AD7F5DFCC82B"
        },
        {
            "_links": {
                "self": {
                    "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/
d3Id/user2",
                    "templated": false
                }
            },
            "d3Id": "user2",
            "hasExportRight": false,
            "hasMigrationRight": false,
            "idpId": ""
        },
        {
            "_links": {
                "self": {
                    "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/
d3Id/user3",
                    "templated": false
                }
            },
            "d3Id": "user3",
            "hasExportRight": false,
            "hasMigrationRight": false,
            "idpId": ""
        }
    ]

}
```

**Objective: Displaying users with migration permissions**

Request/subroute: **GET https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user?hasMigration-Right**

Sample end point response:

```
{
  "user": [
    {
            "_links": {
                "self": {
                    "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/
d3Id/user1",
                    "templated": false
                }
            },
            "d3Id": "user1",
            "hasExportRight": true,
            "hasMigrationRight": true,
```

```
            "idpId": "7B841E93-EC4E-4790-B9D7-AD7F5DFCC82B"
    }
  ]
}
```

**Objective: Displaying users without migration permissions**

Request/subroute: **GET https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user?hasMigrationRight=false**

Sample end point response:

```
{
  "user": [
    {
            "_links": {
                "self": {
                    "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/
d3Id/user2",
                    "templated": false
                }
            },
            "d3Id": "user2",
            "hasExportRight": false,
            "hasMigrationRight": false,
            "idpId": ""
        },
        {
            "_links": {
                "self": {
                    "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/
d3Id/user3",
                    "templated": false
                }
            },
            "d3Id": "user3",
            "hasExportRight": false,
            "hasMigrationRight": false,
            "idpId": ""
        }
  ]
}
```

**Objective: Displaying users with export permissions**

Request/subroute: **GET https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user?hasExportRight**

Sample end point response:

```
{
  "user": [
    {
            "_links": {
                "self": {
                    "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/
d3Id/user1",
```

```
                    "templated": false
                }
            },
            "d3Id": "user1",
            "hasExportRight": true,
            "hasMigrationRight": true,
            "idpId": "7B841E93-EC4E-4790-B9D7-AD7F5DFCC82B"
    }
  ]
}
```

**Objective: Displaying users without export permissions**

Request/subroute: **GET https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user?hasExport-Right=false**

Sample end point response:

```
{
  "user": [
    {
            "_links": {
                "self": {
                    "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/
d3Id/user2",
                    "templated": false
                }
            },
            "d3Id": "user2",
            "hasExportRight": false,
            "hasMigrationRight": false,
            "idpId": ""
        },
        {
            "_links": {
                "self": {
                    "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/
d3Id/user3",
                    "templated": false
                }
            },
            "d3Id": "user3",
            "hasExportRight": false,
            "hasMigrationRight": false,
            "idpId": ""
        }
  ]
}
```

**Objective: Displaying users with IDP IDs**

Request/subroute: **GET https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user?hasIdpId**

Sample end point response:

```
{
  "user": [
```

```
    {
      "_links": {
        "self": {
          "href": "https://<tenant>.d-velop.cloud/dmsdocs/r/<repoId>/user/
d3Id/user1",
          "templated": false
        }
      },
      "d3Id": "user1",
      "hasExportRight": false,
      "hasMigrationRight": true,
      "idpId": "7B841E93-EC4E-4790-B9D7-AD7F5DFCC82B"
    },
    {
      "_links": {
        "self": {
          "href": "https://<tenant>.d-velop.cloud/dmsdocs/r/<repoId>/user/
d3Id/user2",
          "templated": false
        }
      },
      "d3Id": "user2",
      "hasExportRight": false,
      "hasMigrationRight": false,
      "idpId": "61C15487-3A24-4FC0-B105-F3141114168D"
    },
    .....
  ]
}
```

**Objective: Displaying users without IDP IDs**

Request/subroute: **GET https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user?hasIdpId=false**

Sample end point response:

```
{
  "user": [
    {
      "_links": {
        "self": {
          "href": "https://<tenant>.d-velop.cloud/dmsdocs/r/<repoId>/user/
d3Id/user1",
          "templated": false
        }
      },
      "d3Id": "d3_wfl",
      "hasMigrationRight": false,
      "idpId": ""
    },
    {
      "_links": {
        "self": {
          "href": "https://<tenant>.d-velop.cloud/dmsdocs/r/<repoId>/user/
d3Id/user1",
          "templated": false
```

```
        }
      },
      "d3Id": "jhor",
      "hasMigrationRight": false,
      "idpId": ""
    },
    .....
  ]
}
```

**Objective: Querying individual users**

Request/subroute: **GET https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user/d3Id/user1**

Sample end point response:

```
{
  "_links": {
    "self": {
      "href": "https://<tenant>.d-velop.cloud/dmsdocs/r/<repoId>/user/d3Id/
user1",
      "templated": false
    }
  },
  "d3Id": "user1",
  "hasExportRight": false,
  "hasMigrationRight": true,
  "idpId": "7B841E93-EC4E-4790-B9D7-AD7F5DFCC82B"
}
```

## PUT requests

**Objective: Granting and revoking migration permissions**

Request/subroute: **PUT https://<tenant>.d-velop.cloud/dmsdocs/r/<repository ID>/user/d3Id/<d3Id>**

JSON body for request:

```
{
 "d3Id": "user1",
 "hasExportRight": true,
 "hasMigrationRight": false,
 "idpId": "7B841E93-EC4E-4790-B9D7-AD7F5DFCC82B"

}
```

Sample end point response:

```
{
  "_links": {
     "self": {
       "href": "https://<baseUrl>/dmsdocs/r/<repoUuid>/user/d3Id/user1",
       "templated": false
     }
  },
  "d3Id": "user1",
  "hasExportRight": true,
  "hasMigrationRight": false,
```

```
  "idpId": "7B841E93-EC4E-4790-B9D7-AD7F5DFCC82B"
}
```

### 1.2.7. Configuring user rights for on-premises

To use export API, users require the appropriate permissions.

> **Warning**
> Users thus get the permission to export each document. This permission is valid regardless of any restrictions to document types or compulsory watermarks applied.
>
> It applies to both users of the standalone d.velop documents repo export API and users of the dmsdocs app.

**User permissions for exporting data**

The permissions you require for a full data export differ based on the version of the system to be exported. The standalone application requires a d.3 user for the data export. IDP users cannot be granted the required permissions.

**Current versions**

For the following versions of d.velop documents, users require the special **data export** permission. You can grant the permission from the d.3 administration.

- Annual 2022 Patch 09 and later (from 8.21.18)
- Current 2023.Q2 Patch 03 and later (from 8.32.16)
- Beta 2023/08/10 (from 8.33.1)
- Continuous (cloud) (from 8.36.0)

**Out-of-date versions**

In older systems, there is a check of whether users have the **administrator permission**. You have to use the same user when you start d.velop documents repo export API.

### 1.2.8. Using the d.velop documents repo export client

d.velop documents repo exporter is a client solution that can communicate directly with the export API (cloud and on-premises). d.velop documents repo exporter lets you simplify data exports for your customers.

This is a client application that you can start from the Windows command prompt. You can obtain the client via the d.3 server tools package. The client abstracts the HTTP communication to the export API and loads the configured documents to a definable target directory.

The client application provides the following functions:

- You can set the client using the configuration file and start it with appropriate parameters.
- You can use d.velop logfile viewer (d.3 logview) for logging. You can also specify that messages are logged in a separate text file.

**Prerequisites**:

- The export API is accessible, installed and activated.
- An export directory on which the metadata and the original files can be saved is available. During the export, the documents are saved based on the directory structure (document tree).

### Structure of the “Config.json” file

The configuration file includes the following:

- The user information for the source system (on-premises: user name and password, cloud: repository ID and API key)
- URL at which d.velop documents repo exporter can be accessed.
- Filter for defining the documents to be exported.

> **Note**
> The base address of the export API is logged in the output at the start.

The configuration file is structured as follows:

```
{
  "baseUrl" : "<http://127.0.0.1:8333>",
  "user" : "<exportUser>",
  "password" : "<password>",
  "repoId" : "<uuid>",
  "apiKey" : "<apiKey>",
  "timeout" : "<seconds>",
  "proxyPort" : "<port>",
  "proxyHost" : "<ProxyIp>",
  "filter" : {
      "modifiedAfter": "<2017-11-07T13:27:37.643Z>",
      "modifiedBefore": "<2023-01-01T01:01:01.001Z>",
      "numberOfProcesses": <2>,
      "documentTypesByD3Id": [],
      "documentTypesById": [],
      "docIds": [],
      "batchSize": <200>
  }
}
```

Unless otherwise defined, you can also enter all the parameters from the command prompt in the configuration file.

Below you can find descriptions of the parameters in the configuration file:

| Parameter | Description |
|---|---|
| **user** | User name of the user with permissions for exports<br><br>More information about distributing permissions: Configuring user rights for the cloud |
| **password** | Optional: Password of the user exporting the files |
| **apiKey** | Relevant for the cloud: API key of the user with permissions for exports |
| **timeout** | Optional: Request Timeout<br><br>Default: 300 seconds |
| **filter** | Restricting filter for the data export<br><br>More information: Defining the export filter |

## Starting the client through a Windows command prompt

You can start d.velop documents repo exporter with the Windows command prompt (**cmd**). Add the following command prompt and adapt the values to your requirements:

```
export_client.exe --configFile <pathToConfig.json> <optionale Parameter>
```

The export is performed automatically after the application starts.

Below you can find descriptions of the parameters of the command prompt:

| Parameter | Optional | Description |
|---|---|---|
| **--configFile** | **n** | Path to the configuration file for the client. Must be transferred when the application is started. |
| **--baseUrl** | **n** | Base address of the d.velop documents repo exporter API. |
| **--exportPath** | **n** | Path to the target directory for the export (saving the original file and the associated metadata) Must be transferred in the command prompt or configuration file. |
| **--repoId** | **-** | Repository ID of the d.velop documents system to be exported.<br>• Mandatory when using the export API embedded in d.velop documents. This applies to current d.velop documents systems and exports from the cloud.<br>• If you are using the standalone d.velop documents repo export API, you must leave this configuration value out. The d.velop documents system to be exported is defined using the d.velop documents repo exporter API start parameters. |
| **--exportWithFileType** | **n** | This configuration enables exports retaining known file extensions. This mode is not suitable for subsequent migrations or imports to d.velop documents. |
| **--exportMetadataOnly** | **n** | Only the metadata from d.velop documents is exported. This mode can be used to analyze metadata, for example. |
| **--exportRepositoryConfigOnly** | **n** | The client requests only the configuration of the d.velop documents system to be exported and closes after the configuration is saved. |
| **--logFile (optional)** | **n** | Path and name of a log file.<br>The file contains the whole export log and is added to on an ongoing basis. The file is not rolling and can therefore take up a large amount of storage space.<br>We recommend producing a backup copy of the file before running d.velop documents repo exporter again. |
| **--logLevel (optional)** | **n** | Log level (default: **6**)<br>The values are between **0** and **9** (**0**: no logging, **9**: maximum debug logging). |
| **--proxyHost** | **n** | Host name or IP of an HTTP/HTTPS proxy. Authentication on the proxy is not supported. |
| **--proxyPort** | **n** | Port of an HTTP/HTTPS proxy |
| **--ignoreCertificates** | **n** | Allows the use of invalid and self-signed certificates. |
| **--startLink** | **n** | Use this parameter if a previous export failed and you know the URL of the last successful block. This start parameter bypasses any defined filters. |
| **--help** | **n** | Displays a help text for the start parameters and closes the application. |
| **--useSynchIdFormat** | **-** | This parameter is used for the export of the metadata with a unique property ID. No: **n**, Yes: **y**. |

How to export from the cloud: Enter the relevant values in the configuration file for the parameters **repoId** and **apikey**.

The documents are then saved in the specified folder (**exportPath**) based on the directory structure (document tree), together with the JSON display of the metadata. If you selected the applicable option, the master data is also saved in the specified folder.

## Behavior in the event of errors

Information about exported documents is saved in the files **error_log_<number>.json**. The files are numbered sequentially. The files are created after 1,000,000 documents or after 100 errors. After 100,000 documents, a log is written.

**Example of an error in "error_log_0.json"**:

```
{
 "errorBatches": [
   {
     "filter": {
       "docIds": [
         "P000000005",
         "P000000079",
         "P000009725",
```

```
          "P000009729",
          "P000009731",
          "P000009733",
          "P000009735",
          "P000009737",
          "P000009739",
          "P000009742"
        ]
      },
      "link": "/repo/export?
startDocId=P000000005&batchNumber=0&docIds=P000000005,P000000079,P000009725,
P000009729,P000009731,P000009733,P000009735,P000009737,P000009739,P000009742
&bS=100"
    }
  ]
}
```


**Description**:

- **filter**: When the tool is restarted, you can use this parameter to reexport specific document IDs that were not exported.
- **link**: This parameter is the endpoint that caused the error. You can use the endpoint for debugging with external tools.

### 1.2.9. Export format of the metadata mapping

During each export of a dossier or document, files are generated that all have the same basic name: the document ID.

The file extension indicates the file type:

- The .json file extension indicates metadata.
- All other extensions are original files (e.g. .pdf and .tiff).

There is at least one metadata file with each export (e.g. “M000000001.json”). If the metadata file is available only, it is generally a dossier. A separate file is exported for each document version. For these document versions, the document ID (**fileId**) is used as the file extension. In the metadata file, the actual file extension for this version is described in **physicalVersion.extension**, e.g. “M000000001.1”. Dependent files may also be included whose identifiers (**P1** for PDF and **T1** for TIFF files) are also appended, e.g. “M00000000.1.P1”.

The metadata file contains the following information:

- Common properties
- Advanced properties
- System properties
- Version information on the document and dependent files
- Notes
- Link relations
- Activities

Further information on the structure of the metadata files is provided below. Please note that certain details, such as those relating to mandatory fields, may only apply in the context of the current table, and are not necessarily a prerequisite for the metadata format (StandardDocument).

# Structure of the metadata format (StandardDocument)

## Basic structure

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| docId | string | | Document ID | yes |
| attributesByRepoId | StandardDocument.AttributesByRepoIdEntry | repeated | List of advanced properties of the document. The d.3 internal repository field Id serves as the map key. | no |
| attributesById | StandardDocument.AttributesByIdEntry | repeated | List of advanced properties of the document. The Master-Data-SyncId serves of the repository field serves as the map key. | no |
| documentType | **DocumentType** | | Category of the document or dossier | yes |
| versions | **LogicalVersion** | repeated | Versions of a document (one physical and one logical) | yes |
| systemAttributes | **SystemAttributes** | | System properties | yes |
| notes | **Note** | repeated | Notes of a document | no |
| editor | **UserOrGroup** | | Processing of a document by a user or group. Not mandatory in the **Verification** version status. | no |
| parentDocuments | **LinkedDocuments** | repeated | Dossiers that contain this document. The key is the file extension of the dependent file, which you can derive from **dependent_documents.doc_extension**. | no |
| childDocuments | **LinkedDocuments** | repeated | Documents contained in this dossier. | no |
| docSysVals | **DocSysVals** | repeated | Additional system properties. Typically activated by d.ecs content crawler or on a customer-specific basis in e-mails. | no |

| | | | | |
|---|---|---|---|---|
| history | **HistoryEntry** | repeated | Activities and/or history of the document | no |

## Summary of the content of a metadata file

**Minimal scope**


```
{
    "docId": "M000000002",
    "documentType": {
        "d3Id": "FEHME"
    },
    "versions": [{
        "status": "DOC_STAT_RELEASE",
        "physicalVersion": {
            "fileId": 1,
            "file": {
                "sizeInByte": 49,
            }
        },
        "create": {
            "user": {
                "d3Id": "d3_admin"
            },
            "timestamp": "2017-11-07T13:27:37.643Z"
        },
        "physicalVersion":  {
            "dependentFiles":  {
                "P1":  {
                    "file":  {
                        "fileHash":  "MD5:CrqD7HSwcmDDSX97JJ4Z1Q==",
                        "sizeInByte":  "30066",
                        "d3Hash":  "MD5:8C79851D293719041D2AE45332A8D90C"
                    }
                }
            },
            "extension":  "XLS",
            "ocr":  "Testdatei",
            "file":  {
                "fileHash":  "RIPEMD256:RQbE/
y+Dv09GFU0UFfYaWWEDQD5QBHTdwXTM7pqpRbA=",
                "sizeInByte":  "19456",
                "d3Hash":  "MD5:93FC10D3D7AB24F09603B0761789CFF7"
            },
            "fileId":  1
        }
    }],
    "systemAttributes":  {
        "webPub":  false,
        "dateOverallProc":  "2021-11-04T13:21:37Z",
        "updateCounter":  51,
        "dateUpdFile":  "2021-10-11T12:32:59Z",
        "dateUpdAttrib":  "2021-11-04T13:21:37Z",
        "owner":  {
            "d3Id":  "legalVW"
```

```
        },
        "releasedVersionIsBlocked":  false,
        "number":  "LV00001677",
        "varNumber":  1,
        "text":  [
            "",
            "",
            "",
            ""
        ],
        "verificationDone":  false,
        "filename":  "myFilenameWithoutExtension",
        "dateAccess":  "2021-10-11T12:32:59Z",
        "create":  {
            "user":  {
                "d3Id":  "legalVW"
            },
            "timestamp":  "2021-10-11T12:32:59Z"
        },
        "dateRetention":  "2027-10-20T00:00:00Z",
        "colorCode":  0
    },
    "attributesByRepoId":  {
        "7":  {
            "string":  "Lorem ipsum dolor sit amet"
        },
        "4":  {
            "string":  "consetetur sadipscing elitr"
        },
        "5":  {
            "string":  "sed diam nonumy eirmod tempor invidunt ut labore et
dolore magna aliquyam erat"
        },
        "16":  {
            "strings":  {
                "1":  "Mulitline line 1",
                   "2":  "Mulitline line 2"
            }
        },
        "200":  {
            "date":  "2029-10-10"
        }
    },
    "notes":  [
    {
            "create":  {
                "user":  {
                    "d3Id":  "dvelop"
                },
                "timestamp":  "2025-04-02T09:33:05Z"
            },
            "message":  "Dokument LV00058573 wurde von Benutzer legalVW
gelöscht. Document LV00058573 has been deleted by legalVW"
        }
    ],
```

```
"parentDocuments":  [
    {
        "create":  {
            "user":  {
                        "d3Id":  "dvelop"
            },
            "timestamp":  "2021-10-11T11:49:46Z"
        },
        "linkedDocument":  "M000000001"
    }
],
"childDocuments":  [
    {
        "create":  {
            "user":  {
                    "d3Id":  "dvelop"
            },
            "timestamp":  "2021-10-28T08:03:10Z"
        },
        "linkedDocument":  "M000000003"
    }
],
"history":  [
    {
        "eventName":  "link_to_child_job",
        "details":  [
            {
                "string":  "M000000003",
                "detailName":  "child"
            }
        ],
        "timestamp":  "2021-10-28T10:03:10Z",
        "user":  {
            "d3Id":  "d3_async"
        }
    },
    {
        "eventName":  "link_to_parent_job",
        "details":  [
            {
                "string":  "M000000001",
                "detailName":  "parent"
            }
        ],
        "timestamp":  "2021-10-11T13:49:46Z",
        "user":  {
            "d3Id":  "d3_async"
        }
    },
    {
        "eventName":  "import",
        "details":  [
            {
                "string":  "value for docField2",
                "detailName":  "doc_field[2]"
```

```
                },
                {
                    "string":  "value for docField3",
                    "detailName":  "doc_field[3]"
                }
                ...
                ...
                ...
            ],
            "timestamp":  "2021-10-11T13:49:44Z",
            "user":  {
                "d3Id":  "dvelop"
            }
        },
        {
            "eventName":  "retention_event",
            "timestamp":  "2021-10-12T14:52:05Z",
            "user":  {
                "d3Id":  "d3_server"
            }
        }
    ],
    "docSysVals":  [
        {
            "index":  1,
            "name":  "myCustomDocSysValue",
            "value":  "Value one"
        },
        {
            "index":  2,
            "name":  "myCustomDocSysValue",
            "value":  "Value two"
        }
    ]
}
```

## attributesByRepoId and/or attributesById

Contains the advanced properties. The key contains the ID of the advanced property.

**attributesByRepoId** is specified when you export the standard ID (one to three digits) of the property.

**attributesById** is specified when you export the UUID (36 digits) of the property.

```
    "attributesByRepoId":  {
        "7":  {
            "string":  "Lorem ipsum dolor sit amet"
        },
        "4":  {
            "string":  "consetetur sadipscing elitr"
        },
        "5":  {
            "string":  "sed diam nonumy eirmod tempor invidunt ut labore et
dolore magna aliquyam erat"
        },
        "16":  {
            "strings":  {
```

```
                "1":  "Mulitline line 1",
                   "2":  "Mulitline line 2"
            }
        },
        "200":  {
            "date":  "2029-10-10"
        }
    }
```

### AttributeValue

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| string | string | | Alphanumerical property field | no |
| number | double | | Numerical property field (also for monetary fields) | no |
| date | string | | DatePropertyField | no |
| datetime | google.protobuf.Timestamp | | Date + timestamp of property field | no |
| strings | AttributeValue.StringsEntry | repeated | Alphanumerical multi-value property @key line of the value (1-2000) | no |
| numbers | AttributeValue.NumbersEntry | repeated | Numerical multi-value property | no |
| dates | AttributeValue.DatesEntry | repeated | Date of multi-value property | no |
| datetimes | AttributeValue.DatetimesEntry | repeated | Date + timestamp of multi-value property | no |

## documentType

Contains the category/document type

```
    "documentType": {
        "d3Id": "FEHME"
    }
```

### DocumentType

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| d3Id | string | | d.3 internal ID of the document type (reference to DB column arten_dokumente.kue_dokuart, maximum 5 characters) | yes |
| id | string | | Master-Data-SyncId / UUID (identifier can be obtained from dms-config-app) | no |

## versions

Contains the version information.

```
    "versions": [{
        "status": "DOC_STAT_RELEASE",
        "physicalVersion": {
            ...
```

```
        },
        "create": {
            "user": {
                "d3Id": "d3_admin"
            },
            "timestamp": "2017-11-07T13:27:37.643Z"
        }
    }]
```

## LogicalVersion

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| status | **D3DocStatus** | | **DOC_STAT_PROCESSING**: Processing<br>**DOC_STAT_VERIFICATION**: Verification<br>**DOC_STAT_RELEASE**: Release<br>**DOC_STAT_ARCHIVE**: Archive | yes |
| physicalVersion | **PhysicalVersion** | | Dependent physical version (reference to files in the documents tree). Can be omitted with dossiers (documents without a file). | yes, if category **document** |
| create | **Action** | | Document action: **Created** This action must exist for at least one document. | Yes. Can be derived from forwarded system properties. |
| verify | **Action** | | Document action: **Verified** | no |
| release | **Action** | | Document action: **Released** | no |
| block | **Action** | | Document action: **Blocked** | no |
| archive | **Action** | | Document action: **Archived** | no |
| delete | **Action** | | Document action: **Deleted** | no |
| changeReason | string | | Reason for change | no |

| | | | |
|---|---|---|---|
| externalVersionId | double | External version number of the logical document (not generally available) | no |

### Rules for D3DocStatus and version concept

- The logical versions of the document are specified as an array.
- At least one version must be specified.
- A document has a maximum of one version in the **Verification** status or in the **Edit** status or but no version in both statuses.
  (In detail, this means: A document can either have one version in the **Verification** status or one version in the **Edit** status, but it can never have one version in both the**Verification** status and **Edit** status at the same time.
  There are documents that do not have a version in either the Verification status or the Editing status, however.)
- There can also be only one version in the **Release** status.
- Each version that has ever been released is transferred to the **Archive** status as soon as a new release version is available and is retained.
  Therefore, there can be any number of versions in the **Archive** status. (In the case of d.3 systems prior to v8, there is a limit of 999 documents in the **Archive** status).
- Physical versions reference a specific user file and its dependent files. These are listed below the corresponding logical versions.

| Status | | Possible amount |
|---|---|---|
| Processing | Verification | 0 or 1, either Processing or Verification |
| Release | | 0 or 1 |
| Archive 1 -X | | 0 to x |

**Other important fields for documents in a status: Processing status:** A document with this status is required to have an existing editor, which can be a user or a group.

**Verification status:** A document with the “Verification” status can have an editor, but this is not mandatory.

**In addition, each version of a document requires an Action User and a Timestamp.**

**Where can documents be found in on-premise d3 systems?**

**Document tree under v8:**As of d.3 version 8.0, the following also applies to every document status:

- Master files in the d.3 document tree are named <Document ID>.<File ID> (e.g. a document that has been modified many times, P000000042.4711).
- Dependent files in the d.3 document tree are named <Document ID>.<File ID>.<dependent file extension> (e.g. P000000042.4711.P1).

There are no status-related subfolders; all documents use the **docs.** folder.

This folder contains subfolders named with the first 4 characters of the document ID, each containing a further subfolder with the second 4 characters of the document ID:

docs/P000/0000/P000000042.4711

On-premises after being moved to secondary storage: cached_docs//P000/0000/P000000042.4711

A document version receives its FildeId as soon as it enters the Processing status.

If a file never changes, its location remains the same and the FildeId never changes.

### PhysicalVersion

File information about versions:

```
"versions": [{
    ...
    "physicalVersion":  {
        "dependentFiles":  {
            ...
        },
        "extension":  "XLS",
        "ocr":  "Testdatei",
        "file":  {
            ...
        },
        "fileId":  1
    },
    ...
}],
```

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| fileId | uint32 | | ID of the physical version. References a file in the documents tree (<doc_id>.<file_id>) | yes |
| file | **FileDescription** | | Physical file properties | yes |
| dependentFiles | PhysicalVersion.**DependentFilesEntry** | repeated | Dependent files The key in the map is the ID of the dependent file (RegEx: “[A-Z][0-9]”).<br><br>The key is used for the referencing of the file in the documents tree (<doc_id>.<file_id>.<id>). | no |
| extension | string | | File extension (e.g. DOCX, PDF, TXT) | yes |
| ocr | string | | Optical Character Recognition / Full text for this version. OCR data from the current version is used for the full-text searches. | no |

### DependentFile

Dependent files for a document. Typically used for renditions, signatures, or XML property files.

```
   "versions": [{
       ...
       "physicalVersion":  {
           "dependentFiles":  {
               "P1":  {
                   "file":  {
                       "fileHash":  "MD5:CrqD7HSwcmDDSX97JJ4Z1Q==",
                       "sizeInByte":  "30066",
                       "d3Hash":  "MD5:8C79851D293719041D2AE45332A8D90C"
                   }
               }
           },
           ...
       }
       ...
   }],
```

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| file | **FileDescription** | | | yes |

### FileDescription

Description of the common properties of a file in d.3. This is used both for user files (table files_datentraeger) and dependent files (table dependent_files).

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| sizeInByte | uint64 | | Size of the file in the documents tree (unencrypted size) | yes |
| d3Hash | string | | d.3 internal hash format (DB-column files_datentraeger.hash and/or dependent_files.hash) | no |

| | | | |
|---|---|---|---|
| duplicateIdentifier | string | Reference value for duplicates detection. Typically, the cryptographic hash of the file in the configured format (e.g. SHA-256),<br><br>but can be specified by delivering processes to filter out minor binary variations<br><br>(e.g. same Outlook e-mail in different mailboxes) (DB column files_datentraeger.md5). | no |
| fileHash | string | Cryptographic hash of the file as it appears 1:1 in the documents tree (unencrypted). The hash algorithm used depends<br><br>on the configuration of the repository at the time of import. (DB column files-Datentraeger.file_hash and/or dependent_files.file_hash) | no |

## Action

```
"versions": [{
    ...
    "create": {
        "user": {
            "d3Id": "d3_admin"
        },
        "timestamp": "2017-11-07T13:27:37.643Z"
    },
    ...
}],
```

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|

| | | |
|---|---|---|
| user | **User** | yes |
| timestamp | google.protobuf.Timestamp | yes |

### User

Reference to a d.3 user

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| d3Id | string | | d.3-internal user ID (reference to DB column **benutzer.benutzername**, maximum ten characters) | yes (if **idpId** not available) |
| idpId | string | | User ID from the Identity Provider app | yes (if **d3Id** not available) |

## SystemAttributes

Contains the properties:

```
   "systemAttributes":  {
       "webPub":  false,
       "dateOverallProc":  "2021-11-04T13:21:37Z",
       "updateCounter":  51,
       "dateUpdFile":  "2021-10-11T12:32:59Z",
       "dateUpdAttrib":  "2021-11-04T13:21:37Z",
       "owner":  {
           "d3Id":  "Gordon"
       },
       "releasedVersionIsBlocked":  false,
       "number":  "M000000001",
       "varNumber":  1,
       "text":  [
           "",
           "",
           "",
           ""
       ],
       "verificationDone":  false,
       "filename":  "myFilenameWithoutExtension",
       "dateAccess":  "2021-10-11T12:32:59Z",
       "create":  {
           "user":  {
               "d3Id":  "Gordon"
           },
           "timestamp":  "2021-10-11T12:32:59Z"
       },
       "dateRetention":  "2027-10-20T00:00:00Z",
       "colorCode":  0
   }
```

### SystemAttributes

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| text | string | repeated | Comments field of the document, consisting of exactly four lines**DB column phys_datei.text**) | no |
| dateAccess | google.protobuf.Timestamp | | Date of last read access/download (DB column **phys_datei.dat_letzter_zugr**) | no |
| dateUpdAttrib | google.protobuf.Timestamp | | Date of last user file change (**DB column firmen_spezifisch.last_update_attr**) | no |
| dateOverallProc | google.protobuf.Timestamp | | Overall processing date (DB column **phys_datei.overall_proc_date**) | no |
| dateRetention | google.protobuf.Timestamp | | Expiry date for storage (DB column **phys_datei.retention_date**) | No. If not transmitted, the expiry date is calculated internally based on **dateUpdFile** and the document type. |
| dateUpdFile | google.protobuf.Timestamp | | Date of last user file change (DB column **phys_datei.last_update_file**) | no |
| filename | string | | Filename without extension (the extension is taken from the current physical version, DB column **phys_datei.dateiname**) | yes |
| number | string | | ID of the document in the delivering system (DB column **phys_datei.zeich_nr**) | no |
| varNumber | int32 | | If **number**/**zeich_nr** is not unique (DB column **phys_datei.var_nr**). | no |
| create | **Action** | | (DB columns **phys_datei.datum_einbring** and **phys_datei.konstrukteur**) | The create action user can be derived from the owner, but one of the two must be set.<br><br>The createTimestamp is mandatory. |
| owner | **User** | | physdatei.besitzer | Yes, if create Action user was not transferred. |

| | | | |
|---|---|---|---|
| createProgramm | string | phys_datei.erstell_system | no |
| createProgrammVersion | string | phys_datei.version | no |
| accountable | string | phys_datei.zustaendiger | no |
| webPub | bool | phys_datei.web_published | no |
| factory | string | phys_datei.werk_bez | no |
| depart | string | phys_datei.abteilung | no |
| releasedVersionIsBlocked | bool | Specifies whether the version of the document is in the Release blocked status (**true**) or is regularly released.<br><br>The flag only has an effect if the document has a Release version.<br><br>DB columns phys_datei.frei_o_gesperrt: 'g' -> 'true', 'f' -> 'false' | no |
| verificationDone | bool | Specifies whether the version of the document in the Verification status has already been verified (true) or is still awaiting Verification (false)<br><br>(this flag only has an effect if the document has a Verification version).<br><br>DB columns phys_datei.plan_geprueft: 'j' -> 'true', 'n' -> 'false' | no |

## Action

```
"systemAttributes":  {
    ...
    "create":  {
        "user":  {
            "d3Id":  "legalVW"
        },
        "timestamp":  "2021-10-11T12:32:59Z"
    },
    ...
}
```

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| user | **User** | | | yes |
| timestamp | google.protobuf.Timestamp | | | yes |

### User

Reference to a d.3 user.

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| d3Id | string | | d.3 internal user ID (reference to DB column benutzer.benutzername, maximum 10 characters) | yes (if **idpId** not available) |
| idpId | string | | User ID from the Identity Provider app | yes (if **d3Id** not available) |

## notes

Contains the notes

```
"notes": [
{
        "create": {
            "user": {
                "d3Id": "dvelop"
            },
            "timestamp": "2025-04-02T09:33:05Z"
        },
        "message": "Dies ist eine Notiz"
    }
]
```

### Note

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| create | **Action** | | Create event of the note | no |
| message | string | | Note | yes |

### Action

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| user | **User** | | | yes |
| timestamp | google.protobuf.Timestamp | | | yes |

### User

Reference to a d.3 user

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| d3Id | string | | d.3 internal user ID (reference to DB column benutzer.benutzername, maximum 10 characters) | yes (if **idpId** not available) |

| | | | |
|---|---|---|---|
| idpId | string | ID of the user from the Identity Provider app | yes (if **d3Id** not available) |

## parentDocuments/childDocuments

Contains link relations:

```
    "parentDocuments":  [
        {
            "create":  {
                "user":  {
                            "d3Id":  "dvelop"
                },
                "timestamp":  "2021-10-11T11:49:46Z"
            },
            "linkedDocument":  "M000000001"
        }
    ],
    "childDocuments":  [
        {
            "create":  {
                "user":  {
                        "d3Id":  "dvelop"
                },
                "timestamp":  "2021-10-28T08:03:10Z"
            },
        "linkedDocument":  "M000000003"
    }
```

### LinkedDocuments

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| linkedDocument | string | | DocId of the linked item | yes |
| create | **Action** | | dokumenten_verknuepf.link_user + dokumenten_verknuepf.link_tstamp | no |

### Action

```
            "create":  {
                "user":  {
                        "d3Id":  "dvelop"
                },
                "timestamp":  "2021-10-28T08:03:10Z"
            }
```

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| user | **User** | | | yes |
| timestamp | google.protobuf.Timestamp | | | yes |

### User

Reference to a d.3 user.

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| d3Id | string | | d.3 internal user ID (reference to DB column **benutzer.benutzername**, maximum ten characters) | yes (if **idpId** not available) |
| idpId | string | | User ID from the Identity Provider app | yes (if **d3Id** not available) |

## docSysVals

Contains additional system properties. Typically activated by d.ecs content crawler or on a customer-specific basis in e-mails.

```
"docSysVals": [
    {
        "index": 1,
        "name": "myCustomDocSysValue",
        "value": "Value one"
    },
    {
        "index": 2,
        "name": "myCustomDocSysValue",
        "value": "Value two"
    }
]
```

### DocSysVals

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| name | string | | | no |
| index | int32 | | | no |
| value | string | | | no |

## history

### HistoryEntry

Contains the activities:

```
"history": [
    {
        "eventName": "link_to_child_job",
        "details": [
            {
                "string": "M000000003",
                "detailName": "child"
            }
        ],
        "timestamp": "2021-10-28T10:03:10Z",
        "user": {
            "d3Id": "d3_async"
        }
    },
    {
        "eventName": "link_to_parent_job",
        "details": [
            {
```

```
                "string":  "M000000001",
                "detailName":  "parent"
            }
        ],
        "timestamp":  "2021-10-11T13:49:46Z",
        "user":  {
            "d3Id":  "d3_async"
        }
    },
    {
        "eventName":  "import",
        "details":  [
            {
                "string":  "value for docField 2",
                "detailName":  "doc_field[2]"
            },
            {
                "string":  "value for docField 3",
                "detailName":  "doc_field[3]"
            }
            ...
            ...
            ...
        ],
        "timestamp":  "2021-10-11T13:49:44Z",
        "user":  {
            "d3Id":  "dvelop"
        }
    },
    {
        "eventName":  "retention_event",
        "timestamp":  "2021-10-12T14:52:05Z",
        "user":  {
            "d3Id":  "d3_server"
        }
    }
]
```


| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| eventName | string | | A new ID is assigned based on the name. If the ID does not exist, the ID is taken from **doc_history_event**. | yes |
| timestamp | google.protobuf.Timestamp | | | yes |
| user | User | | d3Id or Idpd Id of the user. Table **doc_history Column user_id**. | yes |
| details | HistoryDetails | repeated | | no |

### User

Reference to a d.3 user.

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| d3Id | string | | d.3-internal user ID (reference to DB column **benutzer.benutzername**, maximum ten characters) | yes (if **idpId** not available) |
| idpId | string | | User ID from the Identity Provider app | yes (if **d3Id** not available) |

## HistoryDetails

| Field | Type | Label | Description | Mandatory |
|---|---|---|---|---|
| detailName | string | | A new ID is assigned based on the name. If the ID does not exist, the ID is taken from **doc_history_detail_name**. | yes |
| string | string | | | no |
| numeric | double | | | no |
| integer | uint64 | | | no |
| datetime | google.protobuf.Timestamp | | | no |