Binder Documentation

Release 0.8.0 beta

Artefactual Systems Inc.

February 23, 2016

Contents

CHAPTER 1

User manual

Below you'll find an outline of the chapters in the user manual, and a set of links to each chapter's main sections.

Back to main Binder documentation page

1.1 What is Binder?

Binder is an open-source web application for managing digital repositories. Binder is particularly adept at supporting the care, management, and preservation of complex digital collections such as time-based media and born- digital artworks. The app provides users with a central interface through which they can access, view and manage the rich technical metadata contained in Archival Information Packages (AIPs) held by the repository, as well as managing and describing the relationships between the components of a collections object, its constituent digital objects, and the various external dependencies required to preserve and display the collection over the long-term. Binder gathers together all of this information required to make long-term preservation and assessment decisions in a single user-friendly interface.

Binder integrates and enhances functionality from two existing open source preservation and access applications:

- Archivematica, an open-source digital preservation system taht is designed to maintain standards-based, long-term access to collections of digital objects
- AtoM (Access to Memory), an open-source, web-based application for standards-based description and access

Binder has also been integrated with The Museum System (TMS), and can pull in artwork metadata via the TMS API developed by Steve Moore at the Museum of Modern Art (https://github.com/smoore4moma/TmsApi).

1.1.1 Why Binder?

n. Something that produces or promotes cohesion in loosely assembled substances

- 14. A chemical that causes two substances to bond into one
- 14. A cover or holder for unbound papers, pages, etc
- 14. A plant whose growth habits prevent erosion
- 14. A software mechanism that performs binding (e.g. data binding)
- 14. One who binds.

1.1.2 What can you do with Binder?

Binder aims to support standards-based digital repository management by providing users with a single place to view administrative, technical, descriptive, and preservation metadata related to objects in a repository and the relationships between them. This in turn gives repository managers the information they need to craft appropriate preservation policies and implement decisions for long-term care.

Binder's user-friendly graphical interface provides useful information about your collection, its components, and the supporting technologies required to preserve and display objects held in the repository. The widget-based dashboard includes at-a-glance collection, ingest, download, and fixity information, while the reports module gives administrators detailed information that can be downloaded in CSV format, or viewed directly in the application. The graph-based context browser included on artwork record pages allows users to visualize the components of a work, and manage relationships between a work's components, its AIPs and the files they contain, and related supporting technologies (such as codecs, operating systems, etc) required for ongoing preservation and access. The digital object viewer includes technical metadata extracted from the METS file generated by Archivematica, a copy of the DIP for easy visual reference, and the ability to download files directly or the entire AIP.

With Binder you can:

- Import AIPs and reference copies of digital objects from Archivematica, and relate them to descriptive metadata imported from TMS or created in Binder.
- Gain at-a-glance collection-wide statistics about fixity, ingest, and use via the widget-based dashboard.
- Relate the components of a work to derived AIPs and any supporting technologies required to preserve and display them in the future, using a node-based graphical user interface.
- View and download an AIP's digital objects and technical metadata.
- Sort search and browse results based on facets drawn from both descriptive and technical metadata, allowing for a high degree of precision and granularity - and then save your search parameters for future re-use.
- Run and manage fixity checks of preserved AIPs, and receive alerts if a fixity check fails.

- Track who downloads digital objects from your repository, and why.
- Compare the descriptive and technical metadata of up to 4 digital objects from an AIP side by side in a graphical user interface.
- Generate and save reports on ingest, fixity, usage, and more.

See also:

Learn more about how the Museum of Modern Art (MoMA) is using Binder:

- *Binder at MoMA: an example use case*

Back to top

1.2 Current project status

Binder's original development was planned by MoMA and Artefactual in the second half of 2013, and carried out from January to June 2014. In the initial development, the application was created specifically for MoMA's primary use cases, and made to work within MoMA's environment, including existing applications already in use at the Museum, such as TMS. MoMA currently uses Binder in production within the Musuem.

Now that the initial development goals have been achieved, both Artefactual and MoMA hope to expand the utility of the project by open sourcing its code and making it available to other developers. We believe that Binder can help a broad set of cultural heritage institutions achieve their long-term preservation goals, and would like to see the Binder project develop into a full-fledged, production-ready, open-source application with its own vibrant community.

In late 2014 and early 2015, initial steps to generalize and open-source the code have been undertaken. MoMA was using a custom branch of both Archivematica and AtoM, and the hope is to get Binder functioning/integrated with the most recent public releases of Archivematica and AtoM. **Work remains before the application can be used in its present form, however.**

The main two issues can be summed up as such:

- The initial development was done using Elasticsearch 0.9 as the search index. The most recent AtoM releases use ES 1.3, but the upgrade means that some sections of the code will need to be tested and rewritten before the application is usable.
- MoMA had a custom Archivematica branch that could upload to Binder, but we'd like to make Binder work with the most recent public Archivematica release. In the long-term, this means adding an "Upload to Binder" option into the general Archivematica project. Another goal would be the ability to create new Artwork records via the user interface (rather than via Archivematica upload on the custom Binder branch, and metadata pulled from TMS) - then the usual method of inputting a slug into Archivematica could be used. An even simpler short-term workaround might be to create a command-line script that will generate a new artwork record for upload using the existing slug method in Archivematica. Since none of these workarounds have yet been implemented, at present there is no simple way to attach AIPs and DIPs from Archivematica to nodes in Binder.

We have created some installation instructions using Vagrant, so that developers can work with the code. Note that this will **not** lead to a functioning installation at present - but we hope that community developers might help us tackle some of the isues outlined by our developers as part of the installation notes. See them here:

- https://gist.github.com/sevein/e0b1d036721435add3cd

Resolving these issues are our first priority, so we can better expose the project to the cultural heritage community. We have also listed some long-term project ideas, on the following page, *Project goals: Binder's long-term vision*.

Why not help us out? If you are a developer interested in working with Binder, check out our GitHub repository, and feel free to post questions in our User Forum.

1.3 Project goals: Binder's long-term vision

Artefactual believes that Binder can help cultural institutions meet the challenges of long-term preservation by providing access to metadata (technical, administrative, descriptive, and preservation) that is critical in informing digital preservation policy and practice.

We hope to see the Binder project grow, and are interested in partnering with other cultural heritage institutions looking to solve similar challenges. Below are some of the priority goals we see for the project as it develops.

Important: Please review the *Current project status* first - the below goals are proposals for Binder's long-term development. Our immediate goals are described on the previous page.

1.3.1 Generalize to support a broad set of use cases

Binder was originally developed to help MoMA meet its goals implementing standards-informed, best-practice digital preservation policies and practices. As such, in its present form the application is not ready for broad use in other institutional settings without further development.

Our first development goal is to generalize the application, so it can easily be used to support a broad set of use cases and institutional practices out of the box. In any future development project, we hope to build this kind of generalization into the work carried out.

1.3.2 Remove AtoM as a dependency for Binder

In the initial development of Binder, the app was built by leveraging existing AtoM functionality and adding new features via AngularJS (1.2) front-end pages, which Binder communicates with via an HTTP API. See the *Technical overview* for more information.

The use of AtoM as a backend dependency was a development strategy necessary to meet the desired goals with the resources available. As the project grows, we hope to see Binder gain its own stack (database, web server, search index, etc) tailored specifically to the goals of the project. This will not only improve the long-term maintainability of the project, it will also improve Binder's scalability and, by implementing a common data model already in use by other open-source tools, help to support better interoperability and integration with existing resources used by the cultural heritage community.

1.3.3 Integrate with more existing open-source tools

Binder's strength is in its ability to pull data from several disparate applications used in preservation and conservation, and bring that information together into a user-friendly interface. As the project develops, we would like to see this approach maintained and expanded, so that Binder will be able to communicate with a wide source of existing open source preservation tools, including Islandora, Hydra, BitCurator, and more. Integration with CollectionSpace and other open source alternatives to TMS are also high on our priority list. In general, Artefactual prefers to collaborate with and participate in many of the exciting community development efforts already underway, rather than compete by duplicating the functionality of existing projects in novel ways.

1.3.4 Implement standards-based preservation of Binder data

At present, many of the complex relationships that can be created and managed in Binder are only preserved in the MySQL database that Binder inherits from AtoM. We would like to see those relationships - between AIPs, artworks, components, and supporting technologies - preserved in a standards-based way. This could mean greater integration with Archivematica, to allow for AIP re-ingest or versioning, and/or the creation of Archival Information Collections (AICs) designed to preserve relationships between preserved objects and packages in an OAIS-compliant manner.

Further descriptive metatada import/export functionality should also be considered to support the management of descriptive and technical metadata, the documentation of process history, provenance, and more. Existing linked data ontologies should be explored for their utlity in both preservation and interoperability.

Artefactual has always aimed to support standards-based description and preservation via its AtoM and Archivematica projects - we look forward to continuing and broadening this work via the Binder project.

1.3.5 Develop new tools, reports, and visualizations

Binder's strength is in the access to collections data and metadata that it provides, and as the project grows, we hope to enhance the utility of the application by expanding these capabilities to support more use cases and functions. Below are a few areas where we are excited to consider how best to enhance Binder.

Reports

Presently, Binder includes a number of reports that can be generated by repository managers, including reports on fixity, ingest, download, significant properties, and more. Reports can be viewed in HTML tables via the user interface, or downloaded in CSV format.

Adding further configuration options to existing reports, as well as expanding the number and kind of reports available, are both prime goals for Binder as the application grows.

Dashboard

Binder's dashboard currently includes a number of widgets providing analytics on collection size, growth, character, and activity. Making these widgets configurable, adding further options, and allowing users to better interact with the data displayed (including the ability to download and reuse it) would vastly increase the utility of the Binder dashboard across a broad set of use cases.

Context browser

One of Binder's most striking features is the context browser, available on Artwork records. Built using the D3 library, it provides users with a graphical user interface to create and manage complex relationships between a work, its components, related AIPs, and the supporting technologies required for ongoing preservation and access. By providing this information visually, repository managers have a powerful and intuitive way to explore and maintain collections data.

With the context browser, we have only begun to scratch the surface of what is possible. We see great potential for optimizing and enhancing the existing functionality, to provide repository managers with better and more intuitive tools. Adding the ability to view the current tree in different ways could also open up support for other kinds of cultural heritage data, including trees with large hierarchical structures, such as those found in archival description.

Further views could also be added to the context browser - here are but a few possibilities:

- **AIP view:** shift the context browser's focus from artworks to AIPs, and use the tree to display and manage contained directories, files, objects, as well as related supporting technology and component nodes. Submit changes back to Archivematica for AIP versioning and/or re-ingest.
- **Relationships view:** view and manage relationships between one work in the collection and others.
- **Taxonomy manager:** visualize and manage linked data ontologies and taxonomical hierarchies; manage relationships between a term and other node types in the repository.

The goals and ideas outlined above are not exhaustive - they represent some of the possibilities that have arisen during the development of Binder, and areas which we see as important to consider for the long-term utility of the project.

Have you got other ideas? Why not start a discussion in our User Forum? If you're a developer testing out Binder locally and would like to contribute code, we love pull requests! Check out our Github repository here:

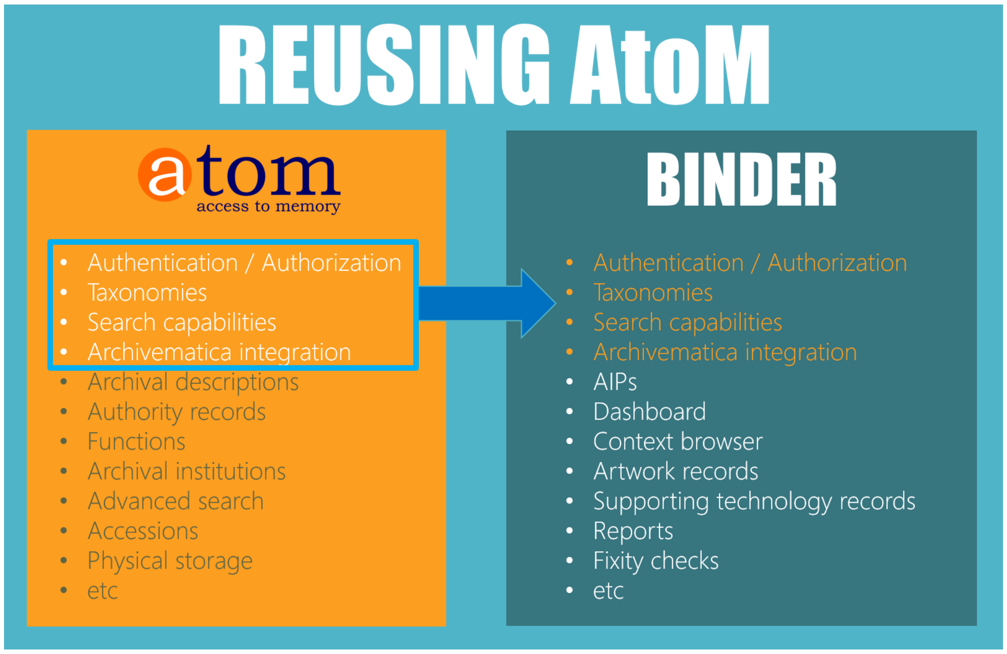
- https://github.com/artefactual/binder

If you are interested in discussing Binder development with Artefactual, please feel free to contact us at info@artefactual.com

Back to top

1.4 Technical overview

Binder has been created by leveraging existing functionality from AtoM, and adding new functionality on top of it.



AtoM is comprised of:

- HTML pages served to a web browser from a web server (Nginx)
- A database on a database server (MySQL 5.1+)
- PHP (5.3.10+) and several PHP extensions, and the Symfony (1.4) framework
- Elasticsearch, a distributed search server based on Apache Lucene, which AtoM interacts with via a RESTful API

For more information, see the AtoM documentation.

Additional modules specific to Binder have been added via AngularJS (1.2) front-end pages, which Binder communicates with via an HTTP API.

Binder-specific UI components, written in AngularJS, have been abstracted from AtoM, which is currently used as a back-end dependency. This abstraction will allow AtoM to be replaced in the future if desired, as Binder grows and the need for a customized environment (e.g. a database, web server, and search index specific to Binder) becomes more relevant.

Since Binder's front-end consumes data via a RESTful API, it can be integrated with other applications in the future with minimal structural changes. See the *API documentation* for more details.

Back to top

1.5 Binder at MoMA: an example use case

Add content here.

Back to top

CHAPTER 2

Administration manual

Below you'll find an outline of the chapters in the administration manual, and a set of links to each chapter's main sections.

Back to main Binder documentation page

2.1 REST API

The Binder user interface is largely written in JavaScript (primarily AngularJS 1.2), running in the browser. The JavaScript then makes requests to REST APIs to create, read, update, and delete data.

Below you'll find documentation for and example usages of the available endpoints:

2.1.1 Activity REST API

The activity REST API endpoint provides data about recent system activity.

Download Activity REST API

The download activity REST API endpoint provides data about recent downloads.

GET /api/activity/downloads
Summary of download activity.

Example request:

```
GET /api/activity/downloads HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: text/javascript

{
    "results": [{
        "date": "2014-06-20 14:38:41",
        "username": "demo",
        "reason": "Reviewing file",
        "file": "0239.mpg"
    }, {
        "date": "2014-06-20 14:37:16",
        "username": "demo",
        "reason": "Comparing file to offline copy",
        "file": "cats.mpg"
    }]
}
```

Query Parameters

- **`limit`** – number of downloads to return

Status Codes

- 200 OK – no error

Back to API documentation index

Ingestion Activity REST API

TODO: add example response

The ingestion activity REST API endpoint provides data about recent ingestions.

`GET /api/activity/ingestion`
Summary of ingestion activity.

Example request:

```
GET /api/activity/ingestion HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: text/javascript

{
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Back to API documentation index

2.1.2 Actor REST API

The actors REST API endpoint provides data about actors that exist in the system.

`GET /api/actors`
Summary of actor data.

Example request:

```
GET /api/actors HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: text/javascript

{
    "results": {
        "621": {
            "updatedAt": "2014-06-23T14:37:59Z",
            "i18n": {
                "languages": ["en"],
                "en": {
                    "authorizedFormOfName": "Rick LaRue"
                }
            },
            "slug": "rick-larue",
            "sourceCulture": "en",
            "createdAt": "2014-06-23T14:37:59Z",
            "authorized_form_of_name": "Rick LaRue"
        }
    },
    "facets": [],
    "total": 1
}
```

Query Parameters

- **`sort`** – field to sort on
- **`sort_direction`** – sort direction, either `asc` (ascending) or `desc` (descending)
- **`limit`** – number of actors to return
- **`skip`** – number of actors to skip (an offset in other words)

Status Codes

- 200 OK – no error

Back to API documentation index

2.1.3 AIPs REST API

The AIP-related REST endpoint allow AIP data to be accessed and AIPS to be reclassified.

Browse AIPs

The AIPs browse REST API endpoint provides data about AIPs that have been added to the system.

`GET /api/aips`

Summary of AIP data.

Example request:

```
GET /api/aips HTTP/1.1
Host: example.com
```

Example response:

```
HTTP/1.1 200 OK
Content-Type: text/javascript

{
    "results": [{
        "id": "417",
        "name": "Electro",
        "uuid": "76e6b3ed-1ba6-403b-92cd-5d3169ca86d2",
        "size": "9078374",
        "created_at": "2014-06-19T15:37:31Z",
        "type": {
            "id": "179",
            "name": "Artwork component"
        },
        "part_of": {
            "id": "416",
            "title": "H49evftydQHN0WidLP7t",
            "level_of_description_id": "375"
        },
        "digital_object_count": "1"
    }, {
        "id": "457",
        "name": "Robert Hunter",
        "uuid": "56a90e59-cab8-496d-8573-f4a3d0588def",
        "size": "646162",
        "created_at": "2014-06-19T15:37:32Z",
        "type": {
            "id": "179",
            "name": "Artwork component"
        },
        "part_of": {
            "id": "456",
            "title": "SWEMJ9EnvX8rx1X5SWA7",
            "level_of_description_id": "375"
        },
        "digital_object_count": "1"
    }],
    "facets": {
        "format": {
            "_type": "terms",
            "missing": 63,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "videoCodec": {
            "_type": "terms",
            "missing": 63,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "audioCodec": {
```

```
                    "_type": "terms",
                    "missing": 63,
                    "total": 0,
                    "other": 0,
                    "terms": []
                },
                "resolution": {
                    "_type": "terms",
                    "missing": 63,
                    "total": 0,
                    "other": 0,
                    "terms": []
                },
                "chromaSubSampling": {
                    "_type": "terms",
                    "missing": 63,
                    "total": 0,
                    "other": 0,
                    "terms": []
                },
                "colorSpace": {
                    "_type": "terms",
                    "missing": 63,
                    "total": 0,
                    "other": 0,
                    "terms": []
                },
                "sampleRate": {
                    "_type": "terms",
                    "missing": 63,
                    "total": 0,
                    "other": 0,
                    "terms": []
                },
                "bitDepth": {
                    "_type": "terms",
                    "missing": 63,
                    "total": 0,
                    "other": 0,
                    "terms": []
                },
                "type": {
                    "_type": "terms",
                    "missing": 15,
                    "total": 48,
                    "other": 0,
                    "terms": [{
                        "term": 181,
                        "count": 15,
                        "label": "Supporting documentation"
                    }, {
                        "term": 180,
                        "count": 14,
                        "label": "Artwork material"
                    }, {
                        "term": 179,
                        "count": 10,
                        "label": "Artwork component"
                    }, {
                        "term": 182,
                        "count": 9,
                        "label": "Supporting technology"
                    }]
                },
                "size": {
                    "_type": "range",
                    "ranges": [{
                        "to": 512000,
                        "count": 0,
                        "total_count": 0,
                        "total": 0,
                        "mean": 0
                    }, {
                        "from": 512000,
                        "to": 1048576,
                        "count": 1,
                        "min": 646162,
                        "max": 646162,
                        "total_count": 1,
                        "total": 646162,
                        "mean": 646162
                    }, {
                        "from": 1048576,
                        "to": 2097152,
                        "count": 6,
                        "min": 1216364,
                        "max": 1654110,
                        "total_count": 6,
                        "total": 8303738,
                        "mean": 1383956.3333333
                    }, {
                        "from": 2097152,
                        "to": 5242880,
                        "count": 18,
                        "min": 2408608,
                        "max": 5215797,
                        "total_count": 18,
                        "total": 63392940,
                        "mean": 3521830
                    }, {
                        "from": 5242880,
                        "to": 10485760,
                        "count": 23,
                        "min": 5676031,
                        "max": 9666391,
                        "total_count": 23,
                        "total": 182412505,
                        "mean": 7930978.4782609
                    }, {
                        "from": 10485760,
                        "count": 0,
                        "total_count": 0,
                        "total": 0,
                        "mean": 0
                    }]
                },
                "dateIngested": {
                    "_type": "range",
                    "ranges": [{
                        "to": 1371970800000,
                        "to_str": "1371970800000",
                        "count": 0,
                        "total_count": 0,
                        "total": 0,
                        "mean": 0,
                        "label": "Older than a year"
                    }, {
                        "from": 1371970800000,
                        "from_str": "1371970800000",
                        "count": 48,
                        "min": 1403192251000,
                        "max": 1403192254000,
                        "total_count": 48,
                        "total": 67353228102000,
                        "mean": 1403192252125,
                    "label": "From last year"
                }, {
                    "from": 1400828400000,
                    "from_str": "1400828400000",
                    "count": 48,
                    "min": 1403192251000,
                    "max": 1403192254000,
                    "total_count": 48,
                    "total": 67353228102000,
                    "mean": 1403192252125,
                    "label": "From last month"
                }, {
                    "from": 1402902000000,
                    "from_str": "1402902000000",
                    "count": 48,
                    "min": 1403192251000,
                    "max": 1403192254000,
                    "total_count": 48,
                    "total": 67353228102000,
                    "mean": 1403192252125,
                    "label": "From last week"
                }]
              }
            },
            "total": 63,
            "overview": {
                "181": {
                    "size": 73896282,
                    "count": 15
                },
                "180": {
                    "size": 77187026,
                    "count": 14
                },
                "179": {
                    "size": 52874675,
                    "count": 10
                },
                "182": {
                    "size": 50797362,
                    "count": 9
                },
                "unclassified": {
                    "count": 15
                },
                "total": {
                    "size": 254755345,
                    "count": 48
                }
            }
        }
```

Query Parameters

- **`sort`** – field to sort on
- **`sort_direction`** – sort direction, either `asc` (ascending) or `desc` (descending)
- **`limit`** – number of AIPs to return
- **`skip`** – number of AIPs to skip (an offset in other words)

Status Codes

- 200 OK – no error

Back to API documentation index

AIP Detail

The AIP detail REST API endpoint provides details about a specific AIP.

`GET /api/aips/<uuid>`
Summary of API data.

Example request:

```
    GET /api/aips/109a8155-9e0e-409c-bb60-b81aca351663 HTTP/1.1
    Host: example.com
    Accept: application/json, text/javascript
```

Example response (METS data removed for the sake of brevity):

```
      HTTP/1.1 200 OK
      Content-Type: text/javascript

    {
        "id": "639",
        "name": "Cat_Pictures",
        "uuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
        "size": "22033310",
        "type": {
            "id": 0,
            "name": null
        },
        "part_of": {
            "id": 622,
            "title": "500",
            "level_of_description_id": 375
        },
        "digital_object_count": "7",
        "digitalObjects": [{
            "id": "640",
            "slug": "0239-mpg-3",
            "identifier": null,
            "inheritReferenceCode": null,
            "levelOfDescriptionId": "386",
            "publicationStatusId": "160",
            "ancestors": ["1", "622", "623", "638"],
            "collectionRootId": 622,
            "parentId": "638",
            "digitalObject": {
                "mediaTypeId": "138",
                "usageId": null,
                "mimeType": "video\/mpeg",
                "byteSize": "4118728",
                "checksum": "b794e283d7f6552717a7311f13bdbedc84945caca7b0203b212c71542a1beb1b",
                "thumbnailPath": null,
                "masterPath": "\/uploads\/r\/null\/b\/7\/b794e283d7f6552717a7311f13bdbedc84945caca7b
            },
            "hasDigitalObject": true,
            "transcript": false,
            "aipUuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aipName": "Cat_Pictures",
            "aipPartOf": "500",
            "aipAttachedTo": "Components",
            "createdAt": "2014-06-23T23:44:45Z",
            "updatedAt": "2014-06-23T23:44:45Z",
            "sourceCulture": "en",
            "i18n": {
                "en": {
                    "title": "0239.mpg"
                },
                "languages": ["en"]
            }
        }, {
            "id": "643",
            "slug": "bigteen-short1-mp3-3",
            "identifier": null,
            "inheritReferenceCode": null,
            "levelOfDescriptionId": "386",
            "publicationStatusId": "160",
            "ancestors": ["1", "622", "623", "638"],
            "collectionRootId": 622,
            "parentId": "638",
            "digitalObject": {
                "mediaTypeId": "135",
                "usageId": null,
                "mimeType": "audio\/mpeg",
                "byteSize": "971275",
                "checksum": "5dc2e020455fdc154c02cb775fbd731ff58c776cf0034f268f7ee3dd7e15f5f2",
                "thumbnailPath": null,
                "masterPath": "\/uploads\/r\/null\/5\/d\/5dc2e020455fdc154c02cb775fbd731ff58c776cf00
            },
            "hasDigitalObject": true,
            "transcript": false,
            "aipUuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aipName": "Cat_Pictures",
            "aipPartOf": "500",
            "aipAttachedTo": "Components",
            "createdAt": "2014-06-23T23:44:48Z",
            "updatedAt": "2014-06-23T23:44:48Z",
            "sourceCulture": "en",
            "i18n": {
                "en": {
                    "title": "BigTeen_Short1.mp3"
                },
                "languages": ["en"]
            }
        }, {
            "id": "646",
            "slug": "blastoff-wmv-3",
            "identifier": null,
            "inheritReferenceCode": null,
            "levelOfDescriptionId": "386",
            "publicationStatusId": "160",
            "ancestors": ["1", "622", "623", "638"],
            "collectionRootId": 622,
            "parentId": "638",
            "hasDigitalObject": false,
            "aipUuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aipName": "Cat_Pictures",
            "aipPartOf": "500",
            "aipAttachedTo": "Components",
            "createdAt": "2014-06-23T23:44:49Z",
            "updatedAt": "2014-06-23T23:44:49Z",
            "sourceCulture": "en",
            "i18n": {
                "en": {
                    "title": "BlastOff.wmv"
                },
                "languages": ["en"]
            }
        }, {
            "id": "647",
            "slug": "funky-breakbeat-4-wav-3",
            "identifier": null,
            "inheritReferenceCode": null,
            "levelOfDescriptionId": "386",
            "publicationStatusId": "160",
            "ancestors": ["1", "622", "623", "638"],
            "collectionRootId": 622,
            "parentId": "638",
            "hasDigitalObject": false,
            "aipUuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aipName": "Cat_Pictures",
            "aipPartOf": "500",
            "aipAttachedTo": "Components",
            "createdAt": "2014-06-23T23:44:49Z",
            "updatedAt": "2014-06-23T23:44:49Z",
            "sourceCulture": "en",
            "i18n": {
                "en": {
                    "title": "funky_breakbeat_4.wav"
                },
                "languages": ["en"]
            }
        }, {
            "id": "648",
            "slug": "j6059-02-wma-3",
            "identifier": null,
            "inheritReferenceCode": null,
            "levelOfDescriptionId": "386",
            "publicationStatusId": "160",
            "ancestors": ["1", "622", "623", "638"],
            "collectionRootId": 622,
            "parentId": "638",
            "hasDigitalObject": false,
            "aipUuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aipName": "Cat_Pictures",
            "aipPartOf": "500",
            "aipAttachedTo": "Components",
            "createdAt": "2014-06-23T23:44:49Z",
            "updatedAt": "2014-06-23T23:44:49Z",
            "sourceCulture": "en",
            "i18n": {
                "en": {
                    "title": "j6059_02.wma"
                },
                "languages": ["en"]
            }
        }, {
            "id": "649",
            "slug": "makeup-mov-3",
            "identifier": null,
            "inheritReferenceCode": null,
            "levelOfDescriptionId": "386",
            "publicationStatusId": "160",
            "ancestors": ["1", "622", "623", "638"],
            "collectionRootId": 622,
            "parentId": "638",
            "hasDigitalObject": false,
            "aipUuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aipName": "Cat_Pictures",
            "aipPartOf": "500",
            "aipAttachedTo": "Components",
            "createdAt": "2014-06-23T23:44:49Z",
            "updatedAt": "2014-06-23T23:44:49Z",
            "sourceCulture": "en",
            "i18n": {
                "en": {
                    "title": "MakeUp.mov"
                },
                "languages": ["en"]
            }
        }, {
            "id": "650",
            "slug": "sample-aif-3",
            "identifier": null,
            "inheritReferenceCode": null,
            "levelOfDescriptionId": "386",
            "publicationStatusId": "160",
            "ancestors": ["1", "622", "623", "638"],
            "collectionRootId": 622,
            "parentId": "638",
            "hasDigitalObject": false,
            "aipUuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aipName": "Cat_Pictures",
            "aipPartOf": "500",
            "aipAttachedTo": "Components",
            "createdAt": "2014-06-23T23:44:50Z",
            "updatedAt": "2014-06-23T23:44:50Z",
            "sourceCulture": "en",
            "i18n": {
                "en": {
                   "title": "sample.aif"
                },
                "languages": ["en"]
            }
        }],
        "created_at": "2014-06-24T06:44:35Z"
    }
```

Status Codes

- 200 OK – no error

Back to API documentation index

AIP Files

The AIP files REST API endpoint provides details about files in a specific AIP.

`GET /api/aips/<uuid>/files`

Summary of API data.

Example request:

```
GET /api/aips/109a8155-9e0e-409c-bb60-b81aca351663/files HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
      HTTP/1.1 200 OK
      Content-Type: text/javascript

    {
        "results": [{
            "id": 646,
            "slug": "blastoff-wmv-3",
            "filename": "BlastOff.wmv",
            "mime_type": "video\/x-ms-asf",
            "byte_size": "1818411",
            "aip_uuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aip_title": "Cat_Pictures"
        }, {
            "id": 640,
            "slug": "0239-mpg-3",
            "filename": "0239.mpg",
            "byte_size": "4118728",
            "media_type_id": "138",
            "master_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/b\/7\/b794e283d7f6552717a7311
            "aip_uuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aip_title": "Cat_Pictures"
        }, {
            "id": 648,
            "slug": "j6059-02-wma-3",
            "filename": "j6059_02.wma",
            "puid": "fmt\/132",
            "mime_type": "video\/x-ms-asf",
            "byte_size": "16418",
            "aip_uuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aip_title": "Cat_Pictures"
        }, {
            "id": 643,
            "slug": "bigteen-short1-mp3-3",
            "filename": "BigTeen_Short1.mp3",
            "puid": "fmt\/134",
            "byte_size": "647296",
            "media_type_id": "135",
            "master_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/5\/d\/5dc2e020455fdc154c02cb7
            "aip_uuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aip_title": "Cat_Pictures"
        }, {
            "id": 647,
            "slug": "funky-breakbeat-4-wav-3",
            "filename": "funky_breakbeat_4.wav",
            "byte_size": "333486",
            "aip_uuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aip_title": "Cat_Pictures"
        }, {
            "id": 649,
            "slug": "makeup-mov-3",
            "filename": "MakeUp.mov",
            "puid": "x-fmt\/384",
            "byte_size": "14955972",
            "aip_uuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aip_title": "Cat_Pictures"
        }, {
            "id": 650,
            "slug": "sample-aif-3",
            "filename": "sample.aif",
            "byte_size": "140834",
            "aip_uuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
            "aip_title": "Cat_Pictures"
        }],
        "total": 7
    }
```

Status Codes

- 200 OK – no error

Back to API documentation index

AIPs Storage Service Status

The AIPs storage service status REST API endpoint provides a list of AIP UUIDs for AIPs that have a specified storage service status.

GET /api/aips/status

Summary of API data.

Example request:

```
GET /api/aips/status?status=RECOVER_REQ HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
HTTP/1.1 200 OK
Content-Type: text/javascript

{
    "uuids": [
        "1b43ab7a-983b-47b3-953f-f026fa090490"
    ]
}
```

Query Parameters

- **status** – storage service status

Status Codes

- 200 OK – no error

Back to API documentation index

AIP Download Proxy

The AIP download REST API endpoint allows the downloading of specific AIP or a file contained in an AIP.

Files are proxied from the storage service.

GET /api/aips/<uuid>/download

Returns file data.

Example request:

```
GET /api/aips/109a8155-9e0e-409c-bb60-b81aca351663/download HTTP/1.1
Host: example.com
```

Query Parameters

- **file_id** – ID of file to download (optional)

Status Codes

- 200 OK – no error

Back to API documentation index

Reclassifiy AIP

The AIP reclassifiy REST API endpoint can be used to reclassify an AIP.

POST /api/aips/<uuid>/reclassify

Summary of API data.

Example request:

```
    POST /api/aips/109a8155-9e0e-409c-bb60-b81aca351663/reclassify HTTP/1.1
    Host: example.com
    Accept: application/json, text/javascript

  {
      "type_id": 180
  }
```

Example response:

```
    HTTP/1.1 200 OK
    Content-Type: text/javascript

  {
      "status": "Saved"
  }
```

Status Codes

- 200 OK – no error

Back to API documentation index

Recover AIP

The AIP recover REST API endpoint can be used to request an AIP be recovered by the storage service.

The recovered AIP files must be placed in the designated recovery directory accessible to the storage service. A storage service administrator will then have to approve the recover request.

POST /api/aips/<uuid>/recover

Summary of API data.

Example request:

```
  POST /api/aips/109a8155-9e0e-409c-bb60-b81aca351663/recover HTTP/1.1
  Host: example.com
```

Example response:

```
    HTTP/1.1 200 OK
    Content-Type: text/javascript

  {
      "message": "Recover request created successfully."
  }
```

Status Codes

- 200 OK – no error

Back to API documentation index

Back to API documentation index

2.1.4 Country REST API

The countries REST API endpoint provides a list of country codes and their corresponding country names.

GET /api/countries
List of countries.

Example request:

```
GET /api/countries HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: text/javascript

{
    "AF": "Afghanistan",
    "AX": "\u00c5land Islands",
    "AL": "Albania",
    "DZ": "Algeria",
    "AS": "American Samoa",
    "AD": "Andorra",
    "AO": "Angola",
    "AI": "Anguilla",
    "AQ": "Antarctica",
    "AG": "Antigua and Barbuda",
    "AR": "Argentina",
    "AM": "Armenia",
    "AW": "Aruba",
    "AU": "Australia",
    "AT": "Austria",
    "AZ": "Azerbaijan",
    "BS": "Bahamas",
    "BH": "Bahrain",
    "BD": "Bangladesh",
    "BB": "Barbados",
    "BY": "Belarus",
    "BE": "Belgium",
    "BZ": "Belize",
    "BJ": "Benin",
    "BM": "Bermuda",
    "BT": "Bhutan",
    "BO": "Bolivia",
    "BA": "Bosnia and Herzegovina",
    "BW": "Botswana",
    "BV": "Bouvet Island",
    "BR": "Brazil",
    "IO": "British Indian Ocean Territory",
    "VG": "British Virgin Islands",
    "BN": "Brunei",
    "BG": "Bulgaria",
    "BF": "Burkina Faso",
    "BI": "Burundi",
    "KH": "Cambodia",
    "CM": "Cameroon",
    "CA": "Canada",
    "CV": "Cape Verde",
    "KY": "Cayman Islands",
    "CF": "Central African Republic",
    "TD": "Chad",
    "CL": "Chile",
    "CN": "China",
    "CX": "Christmas Island",
    "CC": "Cocos [Keeling] Islands",
    "CO": "Colombia",
    "KM": "Comoros",
    "CG": "Congo - Brazzaville",
    "CD": "Congo - Kinshasa",
    "CK": "Cook Islands",
    "CR": "Costa Rica",
    "CI": "C\u00f4te d\u2019Ivoire",
    "HR": "Croatia",
    "CU": "Cuba",
    "CY": "Cyprus",
    "CZ": "Czech Republic",
    "DK": "Denmark",
    "DJ": "Djibouti",
    "DM": "Dominica",
    "DO": "Dominican Republic",
    "EC": "Ecuador",
    "EG": "Egypt",
    "SV": "El Salvador",
    "GQ": "Equatorial Guinea",
    "ER": "Eritrea",
    "EE": "Estonia",
    "ET": "Ethiopia",
    "QU": "European Union",
    "FK": "Falkland Islands",
    "FO": "Faroe Islands",
    "FJ": "Fiji",
    "FI": "Finland",
    "FR": "France",
    "GF": "French Guiana",
    "PF": "French Polynesia",
    "TF": "French Southern Territories",
    "GA": "Gabon",
    "GM": "Gambia",
    "GE": "Georgia",
    "DE": "Germany",
    "GH": "Ghana",
    "GI": "Gibraltar",
    "GR": "Greece",
    "GL": "Greenland",
    "GD": "Grenada",
    "GP": "Guadeloupe",
    "GU": "Guam",
    "GT": "Guatemala",
    "GG": "Guernsey",
    "GN": "Guinea",
    "GW": "Guinea-Bissau",
    "GY": "Guyana",
    "HT": "Haiti",
    "HM": "Heard Island and McDonald Islands",
    "HN": "Honduras",
    "HK": "Hong Kong SAR China",
    "HU": "Hungary",
    "IS": "Iceland",
    "IN": "India",
    "ID": "Indonesia",
    "IR": "Iran",
    "IQ": "Iraq",
    "IE": "Ireland",
    "IM": "Isle of Man",
    "IL": "Israel",
    "IT": "Italy",
    "JM": "Jamaica",
    "JP": "Japan",
    "JE": "Jersey",
    "JO": "Jordan",
    "KZ": "Kazakhstan",
    "KE": "Kenya",
    "KI": "Kiribati",
    "KW": "Kuwait",
    "KG": "Kyrgyzstan",
    "LA": "Laos",
    "LV": "Latvia",
    "LB": "Lebanon",
    "LS": "Lesotho",
    "LR": "Liberia",
    "LY": "Libya",
    "LI": "Liechtenstein",
    "LT": "Lithuania",
    "LU": "Luxembourg",
    "MO": "Macau SAR China",
    "MK": "Macedonia",
    "MG": "Madagascar",
    "MW": "Malawi",
    "MY": "Malaysia",
    "MV": "Maldives",
    "ML": "Mali",
    "MT": "Malta",
    "MH": "Marshall Islands",
    "MQ": "Martinique",
    "MR": "Mauritania",
    "MU": "Mauritius",
    "YT": "Mayotte",
    "MX": "Mexico",
    "FM": "Micronesia",
    "MD": "Moldova",
    "MC": "Monaco",
    "MN": "Mongolia",
    "ME": "Montenegro",
    "MS": "Montserrat",
    "MA": "Morocco",
    "MZ": "Mozambique",
    "MM": "Myanmar [Burma]",
    "NA": "Namibia",
    "NR": "Nauru",
    "NP": "Nepal",
    "NL": "Netherlands",
    "AN": "Netherlands Antilles",
    "NC": "New Caledonia",
    "NZ": "New Zealand",
    "NI": "Nicaragua",
    "NE": "Niger",
    "NG": "Nigeria",
    "NU": "Niue",
    "NF": "Norfolk Island",
    "MP": "Northern Mariana Islands",
    "KP": "North Korea",
    "NO": "Norway",
    "OM": "Oman",
    "QO": "Outlying Oceania",
    "PK": "Pakistan",
    "PW": "Palau",
    "PS": "Palestinian Territories",
    "PA": "Panama",
    "PG": "Papua New Guinea",
    "PY": "Paraguay",
    "PE": "Peru",
    "PH": "Philippines",
    "PN": "Pitcairn Islands",
    "PL": "Poland",
    "PT": "Portugal",
    "PR": "Puerto Rico",
    "QA": "Qatar",
    "RE": "R\u00e9union",
    "RO": "Romania",
    "RU": "Russia",
    "RW": "Rwanda",
    "BL": "Saint Barth\u00e9lemy",
    "SH": "Saint Helena",
    "KN": "Saint Kitts and Nevis",
    "LC": "Saint Lucia",
    "MF": "Saint Martin",
    "PM": "Saint Pierre and Miquelon",
    "VC": "Saint Vincent and the Grenadines",
    "WS": "Samoa",
    "SM": "San Marino",
    "ST": "S\u00e3o Tom\u00e9 and Pr\u00edncipe",
    "SA": "Saudi Arabia",
    "SN": "Senegal",
    "RS": "Serbia",
    "CS": "Serbia and Montenegro",
    "SC": "Seychelles",
    "SL": "Sierra Leone",
    "SG": "Singapore",
    "SK": "Slovakia",
    "SI": "Slovenia",
    "SB": "Solomon Islands",
    "SO": "Somalia",
    "ZA": "South Africa",
    "GS": "South Georgia and the South Sandwich Islands",
    "KR": "South Korea",
    "ES": "Spain",
    "LK": "Sri Lanka",
    "SD": "Sudan",
    "SR": "Suriname",
    "SJ": "Svalbard and Jan Mayen",
    "SZ": "Swaziland",
    "SE": "Sweden",
    "CH": "Switzerland",
    "SY": "Syria",
    "TW": "Taiwan",
    "TJ": "Tajikistan",
    "TZ": "Tanzania",
    "TH": "Thailand",
    "TL": "Timor-Leste",
    "TG": "Togo",
    "TK": "Tokelau",
    "TO": "Tonga",
    "TT": "Trinidad and Tobago",
    "TN": "Tunisia",
    "TR": "Turkey",
    "TM": "Turkmenistan",
    "TC": "Turks and Caicos Islands",
    "TV": "Tuvalu",
    "UG": "Uganda",
    "UA": "Ukraine",
    "AE": "United Arab Emirates",
    "GB": "United Kingdom",
    "US": "United States",
    "ZZ": "Unknown or Invalid Region",
    "UY": "Uruguay",
    "UM": "U.S. Minor Outlying Islands",
    "VI": "U.S. Virgin Islands",
    "UZ": "Uzbekistan",
    "VU": "Vanuatu",
    "VA": "Vatican City",
    "VE": "Venezuela",
    "VN": "Vietnam",
    "WF": "Wallis and Futuna",
    "EH": "Western Sahara",
    "YE": "Yemen",
    "ZM": "Zambia",
    "ZW": "Zimbabwe"
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

2.1.5 Information Objects REST API

The information object-related REST endpoints allow information object data to be accessed and modified.

Browse/Create Information Objects

The information objects browse/create REST API endpoint provides data about information objects that exist in the system and allows new ones to be created.

GET /api/informationobjects

Summary of information object data.

Example request:

```
GET /api/informationobjects HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "results": {
        "422": {
            "updatedAt": "2014-06-19T15:37:31Z",
            "parentId": "1",
            "hasDigitalObject": false,
            "sourceCulture": "en",
            "createdAt": "2014-06-19T15:37:31Z",
            "ancestors": [
                "1"
            ],
            "i18n": {
                "languages": [
                    "en"
                ],
                "en": {
                    "title": "EYfRJFWhiIb6zCdWQoyI"
                }
            },
            "publicationStatusId": "159",
            "levelOfDescriptionId": "375",
            "slug": "eyfrjfwhiib6zcdwqoyi",
            "id": 422,
            "title": "EYfRJFWhiIb6zCdWQoyI"
        },
        "426": {
            "updatedAt": "2014-06-19T15:37:31Z",
            "parentId": "1",
            "hasDigitalObject": false,
            "sourceCulture": "en",
            "createdAt": "2014-06-19T15:37:31Z",
            "ancestors": [
                "1"
            ],
            "i18n": {
                "languages": [
                    "en"
                ],
                "en": {
                    "title": "Wd2yzFfOKsC5oBWsDU4e"
                }
            },
            "publicationStatusId": "159",
            "levelOfDescriptionId": "375",
            "slug": "wd2yzffoksc5obwsdu4e",
            "id": 426,
            "title": "Wd2yzFfOKsC5oBWsDU4e"
        }
    },
    "facets": [ ],
    "total": 101
}
```

Query Parameters

- **sort** – field to sort on
- **sort_direction** – sort direction, either `asc` (ascending) or `desc` (descending)
- **limit** – number of information objects to return
- **skip** – number of information objects to skip (an offset in other words)

Status Codes

- 200 OK – no error

POST /api/informationobjects

Create an information object.

Example request:

```
      POST /api/informationobjects HTTP/1.1
      Host: example.com

    {
        "title": "Book of Cats",
        "description": "This is a book about cats."
    }
```

Example response:

```
      HTTP/1.1 200 OK
      Content-Type: application/json

    {
        "id": 652,
        "parent_id":1
    }
```

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object Detail

The information object detail REST API endpoint provides data about information objects that exist in the system and allows modification and deletion.

GET /api/informationobjects/<id>

Summary of information object data.

Example request:

```
GET /api/informationobjects/528 HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "id": 528,
    "level_of_description_id": 376,
    "parent_id": 527,
    "parent": "Play Dead; Real Time",
    "title": "MoMA 2012"
}
```

Status Codes

- 200 OK – no error

`PUT /api/informationobjects/<id>`

Update information object data.

Example request:

```
  PUT /api/informationobjects/528 HTTP/1.1
  Host: example.com

{
    "title": "Play Dead"
}
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "id":528,
    "parent_id":527
}
```

Status Codes

- 200 OK – no error

`DELETE /api/informationobjects/<id>`

Delete information object.

Example request:

```
DELETE /api/informationobjects/528 HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

null
```

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object Move

The information object move REST API endpoint allows information objects to be moved.

POST /api/informationobjects/<id>/move

Summary of information object data.

Example request:

```
    POST /api/informationobjects/528/move HTTP/1.1
    Host: example.com

  {
      "parent_id": 234
  }
```

Example response:

```
    HTTP/1.1 200 OK
    Content-Type: application/json

  {
    "id": 528,
    "parent_id": 234
  }
```

Status Codes

- 200 OK – no error

Back to API documentation index

Information Objects Tree

The information object tree-related REST endpoints allow information object tree-related data to be accessed.

Information Object Tree

The information object tree REST API endpoint provides data about the hierarchy associated with a specific information object.

GET /api/informationobjects/<id>/tree<id>

Summary of information object data.

Example request:

```
GET /api/informationobjects/528/tree HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "id": 508,
    "level_of_description_id": 375,
    "title": "UXnfY6wiU4D6VNATtiNx",
    "supporting_technologies_count": 0
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object Tree Associations

TODO: example

The information object tree associations REST API endpoint provides data about the associated associated with a specific information object tree.

GET /api/informationobjects/<id>/tree/associations<id>
Summary of information object data.

Example request:

```
GET /api/informationobjects/528/tree/associations HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

[]
```

Status Codes

- 200 OK – no error

Back to API documentation index

Back to API documentation index

Information Object Files

The information objectsi files REST API endpoint allows information object files data to be browsed.

GET /api/informationobjects/<id>/files
Summary of information object files data.

Example request:

```
GET /api/informationobjects/508/files HTTP/1.1
Host: example.com
```

Example response:

```
    HTTP/1.1 200 OK
    Content-Type: application/json

  {
      "results": [],
      "total":0
  }
```

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object Files Browse

The information objectsi files browse REST API endpoint allows information object files data to be browsed.

GET /api/informationobjects/files
Summary of information object files data.

Example request:

```
GET /api/informationobjects/files HTTP/1.1
Host: example.com
```

Example response:

```
    HTTP/1.1 200 OK
    Content-Type: application/json

  {
      "results": {
          "530": {
              "id": 530,
              "filename": "Picture I",
              "slug": "picture-i",
              "media_type_id": "136",
              "byte_size": "16942",
              "mime_type": "image\/png",
              "thumbnail_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/2\/2\/2225caeda4d3421d
              "master_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/f\/d\/fdf08786cb6a21dd2e9
              "media_type": "Image"
          },
          "534": {
              "id": 534,
              "filename": "Picture II",
              "slug": "picture-ii",
              "media_type_id": "136",
              "byte_size": "17296",
              "mime_type": "image\/png",
              "thumbnail_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/1\/c\/1cc5e8dfe914cb5a
              "master_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/f\/b\/fbb049101bf7cadb199
              "media_type": "Image"
          },
          "538": {
              "id": 538,
              "filename": "Picture III",
              "slug": "picture-iii",
              "media_type_id": "136",
              "byte_size": "17316",
              "mime_type": "image\/png",
              "thumbnail_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/9\/7\/97b75215465838a5
              "master_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/8\/e\/8eeb1be71daf22b5f71
              "media_type": "Image"
          },
          "570": {
              "id": 570,
              "filename": "j6059_02.wma",
              "slug": "j6059-02-wma",
              "media_type_id": "135",
              "byte_size": "37033",
              "size_in_aip": "16418",
              "date_ingested": "2014-06-19T22:38:59Z",
              "mime_type": "audio\/mpeg",
              "thumbnail_path": "http:\/\/10.0.2.15:8003\/images\/.png",
              "master_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/f\/6\/f68c265ff0f44a6d43c
              "media_type": "Audio",
              "original_relative_path_within_aip": "objects\/j6059_02.wma"
          },
          "633": {
              "id": 633,
              "filename": "funky_breakbeat_4.wav",
              "slug": "funky-breakbeat-4-wav-2",
              "size_in_aip": "333486",
              "date_ingested": "2014-06-24T06:20:34Z",
              "thumbnail_path": "http:\/\/10.0.2.15:8003\/images\/.png",
              "master_path": "http:\/\/10.0.2.15:8003\/images\/.png",
              "aip_uuid": "12e4d9eb-b00f-464d-914e-4d6c7b0fe8b6",
              "aip_title": "ABBBBA"
          },
          "646": {
              "id": 646,
              "filename": "BlastOff.wmv",
              "slug": "blastoff-wmv-3",
              "size_in_aip": "1818411",
              "date_ingested": "2014-06-24T06:41:34Z",
              "thumbnail_path": "http:\/\/10.0.2.15:8003\/images\/.png",
              "master_path": "http:\/\/10.0.2.15:8003\/images\/.png",
              "aip_uuid": "d9e5f49c-6d2a-4d01-8e6e-c1d602ff9229",
              "aip_title": "Cat_Pictures"
          },
          "566": {
              "id": 566,
              "filename": "BlastOff.wmv",
              "slug": "blastoff-wmv",
              "media_type_id": "138",
              "byte_size": "2459959",
              "size_in_aip": "1818411",
              "date_ingested": "2014-06-19T22:38:59Z",
              "mime_type": "video\/mp4",
              "thumbnail_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/b\/1\/b12778c40641639d
              "master_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/b\/1\/b12778c40641639d11b
              "media_type": "Video",
              "original_relative_path_within_aip": "objects\/BlastOff.wmv"
              },
              "579": {
                  "id": 579,
                  "filename": "0239.mpg",
                  "slug": "0239-mpg",
                  "media_type_id": "138",
                  "byte_size": "4118728",
                  "size_in_aip": "4118728",
                  "date_ingested": "2014-06-19T22:39:00Z",
                  "mime_type": "video\/mpeg",
                  "thumbnail_path": "http:\/\/10.0.2.15:8003\/images\/.png",
                  "master_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/b\/7\/b794e283d7f6552717a
                  "media_type": "Video",
                  "original_relative_path_within_aip": "objects\/0239.mpg"
              },
              "629": {
                  "id": 629,
                  "filename": "BigTeen_Short1.mp3",
                  "slug": "bigteen-short1-mp3-2",
                  "media_type_id": "135",
                  "byte_size": "971275",
                  "size_in_aip": "647296",
                  "date_ingested": "2014-06-24T06:20:34Z",
                  "mime_type": "audio\/mpeg",
                  "thumbnail_path": "http:\/\/10.0.2.15:8003\/images\/.png",
                  "master_path": "http:\/\/10.0.2.15:8003\/uploads\/r\/null\/5\/d\/5dc2e020455fdc154c0
                  "media_type": "Audio",
                  "aip_uuid": "12e4d9eb-b00f-464d-914e-4d6c7b0fe8b6",
                  "aip_title": "ABBBBA"
              },
              "634": {
                  "id": 634,
                  "filename": "j6059_02.wma",
                  "slug": "j6059-02-wma-2",
                  "size_in_aip": "16418",
                  "date_ingested": "2014-06-24T06:20:34Z",
                  "thumbnail_path": "http:\/\/10.0.2.15:8003\/images\/.png",
                  "master_path": "http:\/\/10.0.2.15:8003\/images\/.png",
                  "aip_uuid": "12e4d9eb-b00f-464d-914e-4d6c7b0fe8b6",
                  "aip_title": "ABBBBA"
              }
          },
          "facets": {
              "format": {
                  "_type": "terms",
                  "missing": 9,
                  "total": 15,
                  "other": 0,
                  "terms": [{
                      "term": "Wave",
                      "count": 3,
                      "label": "Wave"
                  }, {
                      "term": "QuickTime",
                      "count": 3,
                      "label": "QuickTime"
                  }, {
                      "term": "MPEG Video",
                  "count": 3,
                  "label": "MPEG Video"
              }, {
                  "term": "MPEG Audio",
                  "count": 3,
                  "label": "MPEG Audio"
              }, {
                  "term": "AIFF",
                  "count": 3,
                  "label": "AIFF"
              }]
          },
          "videoCodec": {
              "_type": "terms",
              "missing": 18,
              "total": 6,
              "other": 0,
              "terms": [{
                  "term": "jpeg",
                  "count": 3,
                  "label": "jpeg"
              }, {
                  "term": "MPEG-1V",
                  "count": 3,
                  "label": "MPEG-1V"
              }]
          },
          "audioCodec": {
              "_type": "terms",
              "missing": 12,
              "total": 12,
              "other": 0,
              "terms": [{
                  "term": "PCM",
                  "count": 9,
                  "label": "PCM"
              }, {
                  "term": "MPA1L3",
                  "count": 3,
                  "label": "MPA1L3"
              }]
          },
          "resolution": {
              "_type": "terms",
              "missing": 21,
              "total": 3,
              "other": 0,
              "terms": [{
                  "term": 8,
                  "count": 3,
                  "label": "8 bits"
              }]
          },
          "chromaSubSampling": {
              "_type": "terms",
              "missing": 24,
              "total": 0,
              "other": 0,
                  "terms": []
              },
              "colorSpace": {
                  "_type": "terms",
                  "missing": 21,
                  "total": 3,
                  "other": 0,
                  "terms": [{
                      "term": "YUV",
                      "count": 3,
                      "label": "YUV"
                  }]
              },
              "sampleRate": {
                  "_type": "terms",
                  "missing": 12,
                  "total": 12,
                  "other": 0,
                  "terms": [{
                      "term": 44100,
                      "count": 6,
                      "label": "44100 Hz"
                  }, {
                      "term": 22050,
                      "count": 3,
                      "label": "22050 Hz"
                  }, {
                      "term": 8000,
                      "count": 3,
                      "label": "8000 Hz"
                  }]
              },
              "bitDepth": {
                  "_type": "terms",
                  "missing": 21,
                  "total": 3,
                  "other": 0,
                  "terms": [{
                      "term": 8,
                      "count": 3,
                      "label": "8 bits"
                  }]
              },
              "size": {
                  "_type": "range",
                  "ranges": [{
                      "to": 512000,
                      "count": 9,
                      "min": 16418,
                      "max": 333486,
                      "total_count": 9,
                      "total": 1472214,
                      "mean": 163579.33333333
                  }, {
                      "from": 512000,
                      "to": 1048576,
                      "count": 3,
                      "min": 647296,
                  "max": 647296,
                  "total_count": 3,
                  "total": 1941888,
                  "mean": 647296
              }, {
                  "from": 1048576,
                  "to": 2097152,
                  "count": 3,
                  "min": 1818411,
                  "max": 1818411,
                  "total_count": 3,
                  "total": 5455233,
                  "mean": 1818411
              }, {
                  "from": 2097152,
                  "to": 5242880,
                  "count": 3,
                  "min": 4118728,
                  "max": 4118728,
                  "total_count": 3,
                  "total": 12356184,
                  "mean": 4118728
              }, {
                  "from": 5242880,
                  "to": 10485760,
                  "count": 0,
                  "total_count": 0,
                  "total": 0,
                  "mean": 0
              }, {
                  "from": 10485760,
                  "count": 3,
                  "min": 14955972,
                  "max": 14955972,
                  "total_count": 3,
                  "total": 44867916,
                  "mean": 14955972
              }]
          },
          "dateIngested": {
              "_type": "range",
              "ranges": [{
                  "to": 1372143600000,
                  "to_str": "1372143600000",
                  "count": 0,
                  "total_count": 0,
                  "total": 0,
                  "mean": 0,
                  "label": "Older than a year"
              }, {
                  "from": 1372143600000,
                  "from_str": "1372143600000",
                  "count": 21,
                  "min": 1403217539000,
                  "max": 1403592094000,
                  "total_count": 21,
                  "total": 29472803268000,
                  "mean": 1403466822285.7,
                      "label": "From last year"
                  }, {
                      "from": 1401001200000,
                      "from_str": "1401001200000",
                      "count": 21,
                      "min": 1403217539000,
                      "max": 1403592094000,
                      "total_count": 21,
                      "total": 29472803268000,
                      "mean": 1403466822285.7,
                      "label": "From last month"
                  }, {
                      "from": 1403074800000,
                      "from_str": "1403074800000",
                      "count": 21,
                      "min": 1403217539000,
                      "max": 1403592094000,
                      "total_count": 21,
                      "total": 29472803268000,
                      "mean": 1403466822285.7,
                      "label": "From last week"
                  }]
              }
          },
          "total": 24
      }
```

Query Parameters

- **`query`** – search text
- **`sort`** – field to sort on
- **`sort_direction`** – sort direction, either `asc` (ascending) or `desc` (descending)
- **`limit`** – number of information object works to return
- **`skip`** – number of information object works to skip (an offset in other words)

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object TMS Data

The information objectsi TMS data REST API endpoint allows information object TMS data to be browsed.

`GET /api/informationobjects/<id>/tms`
Summary of information object TMS data.

Example request:

```
GET /api/informationobjects/508/tms HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "title": "UXnfY6wiU4D6VNATtiNx",
    "type": "Object"
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object Works

The information objectsi works REST API endpoint allows information object works data to be browsed.

GET /api/informationobjects/works

Summary of information object works data.

Example request:

```
GET /api/informationobjects/works HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "results": {
        "422": {
            "title": "EYfRJFWhiIb6zCdWQoyI"
        },
        "426": {
            "title": "Wd2yzFfOKsC5oBWsDU4e"
        },
        "440": {
            "title": "VXDd26vFl9Zi73hjb0Gh"
        },
        "444": {
            "title": "Ij6IDU3JXbaKjqnOKqHr"
        },
        "448": {
            "title": "wlfasaYtaWH8LhTiKO1A"
        },
        "462": {
            "title": "hLabYNOaMHOAEcAvK2f8"
        },
        "466": {
            "title": "WArI0yvajHxuObIBc0Bo"
        },
        "480": {
            "title": "27kq1uayZrlSg7fB0uk5"
        },
        "484": {
                "title": "yUZgbBryjDLyFYJVyt0Q"
            },
            "488": {
                "title": "tCcgTsVYfz8jKibpczDh"
            }
        },
        "facets": {
            "format": {
                "_type": "terms",
                "missing": 52,
                "total": 5,
                "other": 0,
                "terms": [{
                    "term": "Wave",
                    "count": 1,
                    "label": "Wave"
                }, {
                    "term": "QuickTime",
                    "count": 1,
                    "label": "QuickTime"
                }, {
                    "term": "MPEG Video",
                    "count": 1,
                    "label": "MPEG Video"
                }, {
                    "term": "MPEG Audio",
                    "count": 1,
                    "label": "MPEG Audio"
                }, {
                    "term": "AIFF",
                    "count": 1,
                    "label": "AIFF"
                }]
            },
            "videoCodec": {
                "_type": "terms",
                "missing": 52,
                "total": 2,
                "other": 0,
                "terms": [{
                    "term": "jpeg",
                    "count": 1,
                    "label": "jpeg"
                }, {
                    "term": "MPEG-1V",
                    "count": 1,
                    "label": "MPEG-1V"
                }]
            },
            "audioCodec": {
                "_type": "terms",
                "missing": 52,
                "total": 2,
                "other": 0,
                "terms": [{
                    "term": "PCM",
                    "count": 1,
                    "label": "PCM"
            }, {
                "term": "MPA1L3",
                "count": 1,
                "label": "MPA1L3"
            }]
        },
        "resolution": {
            "_type": "terms",
            "missing": 52,
            "total": 1,
            "other": 0,
            "terms": [{
                "term": 8,
                "count": 1,
                "label": "8 bits"
            }]
        },
        "chromaSubSampling": {
            "_type": "terms",
            "missing": 53,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "colorSpace": {
            "_type": "terms",
            "missing": 52,
            "total": 1,
            "other": 0,
            "terms": [{
                "term": "YUV",
                "count": 1,
                "label": "YUV"
            }]
        },
        "sampleRate": {
            "_type": "terms",
            "missing": 52,
            "total": 3,
            "other": 0,
            "terms": [{
                "term": 44100,
                "count": 1,
                "label": "44100 Hz"
            }, {
                "term": 22050,
                "count": 1,
                "label": "22050 Hz"
            }, {
                "term": 8000,
                "count": 1,
                "label": "8000 Hz"
            }]
        },
        "bitDepth": {
            "_type": "terms",
            "missing": 52,
            "total": 1,
                    "other": 0,
                    "terms": [{
                        "term": 8,
                        "count": 1,
                        "label": "8 bits"
                    }]
                },
                "classification": {
                    "_type": "terms",
                    "missing": 53,
                    "total": 0,
                    "other": 0,
                    "terms": []
                },
                "department": {
                    "_type": "terms",
                    "missing": 53,
                    "total": 0,
                    "other": 0,
                    "terms": []
                },
                "dateCollected": {
                    "_type": "range",
                    "ranges": [{
                        "to": 1372057200000,
                        "to_str": "1372057200000",
                        "count": 0,
                        "total_count": 0,
                        "total": 0,
                        "mean": 0,
                        "label": "Older than a year"
                    }, {
                        "from": 1372057200000,
                        "from_str": "1372057200000",
                        "count": 0,
                        "total_count": 0,
                        "total": 0,
                        "mean": 0,
                        "label": "From last year"
                    }, {
                        "from": 1400914800000,
                        "from_str": "1400914800000",
                        "count": 0,
                        "total_count": 0,
                        "total": 0,
                        "mean": 0,
                        "label": "From last month"
                    }, {
                        "from": 1402988400000,
                        "from_str": "1402988400000",
                        "count": 0,
                        "total_count": 0,
                        "total": 0,
                        "mean": 0,
                        "label": "From last week"
                    }]
                },
                "dateCreated": {
```

```
                "_type": "range",
                "ranges": [{
                    "to": 1372057200000,
                    "to_str": "1372057200000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "Older than a year"
                }, {
                    "from": 1372057200000,
                    "from_str": "1372057200000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "From last year"
                }, {
                    "from": 1400914800000,
                    "from_str": "1400914800000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "From last month"
                }, {
                    "from": 1402988400000,
                    "from_str": "1402988400000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "From last week"
                }]
            },
            "dateIngested": {
                "_type": "range",
                "ranges": [{
                    "to": 1372057200000,
                    "to_str": "1372057200000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "Older than a year"
                }, {
                    "from": 1372057200000,
                    "from_str": "1372057200000",
                    "count": 1,
                    "min": 1403591774000,
                    "max": 1403591774000,
                    "total_count": 1,
                    "total": 1403591774000,
                    "mean": 1403591774000,
                    "label": "From last year"
                }, {
                    "from": 1400914800000,
                    "from_str": "1400914800000",
                    "count": 1,
                    "min": 1403591774000,
                    "max": 1403591774000,
                    "total_count": 1,
                    "total": 1403591774000,
                    "mean": 1403591774000,
                    "label": "From last month"
                }, {
                    "from": 1402988400000,
                    "from_str": "1402988400000",
                    "count": 1,
                    "min": 1403591774000,
                    "max": 1403591774000,
                    "total_count": 1,
                    "total": 1403591774000,
                    "mean": 1403591774000,
                    "label": "From last week"
                }]
            },
            "totalSize": {
                "_type": "range",
                "ranges": [{
                    "to": 512000,
                    "count": 52,
                    "min": 0,
                    "max": 0,
                    "total_count": 52,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 512000,
                    "to": 1048576,
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 1048576,
                    "to": 2097152,
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 2097152,
                    "to": 5242880,
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 5242880,
                    "to": 10485760,
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 10485760,
                    "count": 1,
                    "min": 44066614,
                    "max": 44066614,
                    "total_count": 1,
                    "total": 44066614,
                    "mean": 44066614
                }]
            }
        },
        "total": 53
    }
```

Query Parameters

- **query** – search text
- **totalSizeFrom** – total size from
- **totalSizeTo** – total size to
- **sort** – field to sort on
- **sort_direction** – sort direction, either asc (ascending) or desc (descending)
- **limit** – number of information object works to return
- **skip** – number of information object works to skip (an offset in other words)

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object Components

The information object components REST API endpoint allows information object component data to be browsed.

GET /api/informationobjects/components
Summary of information object components data.

Example request:

```
GET /api/informationobjects/components HTTP/1.1
Host: example.com
```

Example response:

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "results": {
        "543": {
            "name": "1098.2005.a.AV",
            "lod_name": "Exhibition format",
            "artwork_id": "527",
            "artwork_title": "Play Dead; Real Time"
        },
        "552": {
                "name": "1098.2005.b.x1",
                "lod_name": "Artist supplied master",
                "artwork_id": "527",
                "artwork_title": "Play Dead; Real Time"
            },
            "556": {
                "name": "1098.2005.c.x2",
                "lod_name": "Artist verified proof",
                "artwork_id": "527",
                "artwork_title": "Play Dead; Real Time"
            },
            "561": {
                "name": "1098.2005.c.x4",
                "lod_name": "Archival master",
                "artwork_id": "527",
                "artwork_title": "Play Dead; Real Time"
            },
            "548": {
                "name": "1098.2005.a.x1",
                "lod_name": "Artist supplied master",
                "artwork_id": "527",
                "artwork_title": "Play Dead; Real Time"
            },
            "553": {
                "name": "1098.2005.b.x2",
                "lod_name": "Artist verified proof",
                "artwork_id": "527",
                "artwork_title": "Play Dead; Real Time"
            },
            "557": {
                "name": "1098.2005.c.x3",
                "lod_name": "Artist verified proof",
                "artwork_id": "527",
                "artwork_title": "Play Dead; Real Time"
            },
            "546": {
                "name": "1098.2005.c.AV",
                "lod_name": "Exhibition format",
                "artwork_id": "527",
                "artwork_title": "Play Dead; Real Time"
            },
            "555": {
                "name": "1098.2005.c.x1",
                "lod_name": "Artist supplied master",
                "artwork_id": "527",
                "artwork_title": "Play Dead; Real Time"
            },
            "559": {
                "name": "1098.2005.a.x4",
                "lod_name": "Archival master",
                "artwork_id": "527",
                "artwork_title": "Play Dead; Real Time"
            }
        },
        "facets": {
            "format": {
                "_type": "terms",
                "missing": 15,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "videoCodec": {
            "_type": "terms",
            "missing": 15,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "audioCodec": {
            "_type": "terms",
            "missing": 15,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "resolution": {
            "_type": "terms",
            "missing": 15,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "chromaSubSampling": {
            "_type": "terms",
            "missing": 15,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "colorSpace": {
            "_type": "terms",
            "missing": 15,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "sampleRate": {
            "_type": "terms",
            "missing": 15,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "bitDepth": {
            "_type": "terms",
            "missing": 15,
            "total": 0,
            "other": 0,
            "terms": []
        },
        "classification": {
            "_type": "terms",
            "missing": 15,
            "total": 0,
            "other": 0,
                "terms": []
            },
            "type": {
                "_type": "terms",
                "missing": 0,
                "total": 15,
                "other": 0,
                "terms": [{
                    "term": 379,
                    "count": 6,
                    "label": "Artist verified proof"
                }, {
                    "term": 381,
                    "count": 3,
                    "label": "Exhibition format"
                }, {
                    "term": 380,
                    "count": 3,
                    "label": "Archival master"
                }, {
                    "term": 378,
                    "count": 3,
                    "label": "Artist supplied master"
                }]
            },
            "dateIngested": {
                "_type": "range",
                "ranges": [{
                    "to": 1372057200000,
                    "to_str": "1372057200000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "Older than a year"
                }, {
                    "from": 1372057200000,
                    "from_str": "1372057200000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "From last year"
                }, {
                    "from": 1400914800000,
                    "from_str": "1400914800000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "From last month"
                }, {
                    "from": 1402988400000,
                    "from_str": "1402988400000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "From last week"
                }]
            },
            "totalSize": {
                "_type": "range",
                "ranges": [{
                    "to": 512000,
                    "count": 15,
                    "min": 0,
                    "max": 0,
                    "total_count": 15,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 512000,
                    "to": 1048576,
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 1048576,
                    "to": 2097152,
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 2097152,
                    "to": 5242880,
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 5242880,
                    "to": 10485760,
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0
                }, {
                    "from": 10485760,
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0
                }]
            }
        },
        "total": 15
    }
```

Query Parameters

- **`query`** – search text
- **`totalSizeFrom`** – total size from

- **totalSizeTo** – total size to
- **sort** – field to sort on
- **sort_direction** – sort direction, either `asc` (ascending) or `desc` (descending)
- **limit** – number of information object components to return
- **skip** – number of information object components to skip (an offset in other words)

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object Technologies

The information object technologies REST API endpoint allows information object technology data to be browsed.

GET /api/informationobjects/technologies

Summary of information object technology data.

Example request:

```
GET /api/informationobjects/technologies HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "results": {
        "512": {
            "title": "Open source",
            "inherited_title": "Codecs \u00bb Video codecs \u00bb Open source",
            "collection_root_id": 510,
            "id": 512
        },
        "516": {
            "title": "Propietary",
            "inherited_title": "Codecs \u00bb Video codecs \u00bb Propietary",
            "collection_root_id": 510,
            "id": 516
        },
        "521": {
            "title": "Open source",
            "inherited_title": "Codecs \u00bb Audio codecs \u00bb Open source",
            "collection_root_id": 510,
            "id": 521
        },
        "525": {
            "title": "WMA",
            "inherited_title": "Codecs \u00bb Audio codecs \u00bb Propietary \u00bb WMA",
            "collection_root_id": 510,
            "id": 525
        },
        "513": {
            "title": "x264",
                "inherited_title": "Codecs \u00bb Video codecs \u00bb Open source \u00bb x264",
                "collection_root_id": 510,
                "id": 513
            },
            "517": {
                "title": "WMV 7",
                "inherited_title": "Codecs \u00bb Video codecs \u00bb Propietary \u00bb WMV 7",
                "collection_root_id": 510,
                "id": 517
            },
            "522": {
                "title": "FLAC",
                "inherited_title": "Codecs \u00bb Audio codecs \u00bb Open source \u00bb FLAC",
                "collection_root_id": 510,
                "id": 522
            },
            "526": {
                "title": "MP3",
                "inherited_title": "Codecs \u00bb Audio codecs \u00bb Propietary \u00bb MP3",
                "collection_root_id": 510,
                "id": 526
            },
            "511": {
                "title": "Video codecs",
                "inherited_title": "Codecs \u00bb Video codecs",
                "collection_root_id": 510,
                "id": 511
            },
            "515": {
                "title": "Xvid",
                "inherited_title": "Codecs \u00bb Video codecs \u00bb Open source \u00bb Xvid",
                "collection_root_id": 510,
                "id": 515
            }
        },
        "facets": {
            "format": {
                "_type": "terms",
                "missing": 17,
                "total": 0,
                "other": 0,
                "terms": []
            },
            "videoCodec": {
                "_type": "terms",
                "missing": 17,
                "total": 0,
                "other": 0,
                "terms": []
            },
            "audioCodec": {
                "_type": "terms",
                "missing": 17,
                "total": 0,
                "other": 0,
                "terms": []
            },
            "resolution": {
                "_type": "terms",
                "missing": 17,
                "total": 0,
                "other": 0,
                "terms": []
            },
            "chromaSubSampling": {
                "_type": "terms",
                "missing": 17,
                "total": 0,
                "other": 0,
                "terms": []
            },
            "colorSpace": {
                "_type": "terms",
                "missing": 17,
                "total": 0,
                "other": 0,
                "terms": []
            },
            "sampleRate": {
                "_type": "terms",
                "missing": 17,
                "total": 0,
                "other": 0,
                "terms": []
            },
            "bitDepth": {
                "_type": "terms",
                "missing": 17,
                "total": 0,
                "other": 0,
                "terms": []
            },
            "dateIngested": {
                "_type": "range",
                "ranges": [{
                    "to": 1372057200000,
                    "to_str": "1372057200000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "Older than a year"
                }, {
                    "from": 1372057200000,
                    "from_str": "1372057200000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                    "mean": 0,
                    "label": "From last year"
                }, {
                    "from": 1400914800000,
                    "from_str": "1400914800000",
                    "count": 0,
                    "total_count": 0,
                    "total": 0,
                "mean": 0,
                "label": "From last month"
            }, {
                "from": 1402988400000,
                "from_str": "1402988400000",
                "count": 0,
                "total_count": 0,
                "total": 0,
                "mean": 0,
                "label": "From last week"
            }]
        },
        "totalSize": {
            "_type": "range",
            "ranges": [{
                "to": 512000,
                "count": 17,
                "min": 0,
                "max": 0,
                "total_count": 17,
                "total": 0,
                "mean": 0
            }, {
                "from": 512000,
                "to": 1048576,
                "count": 0,
                "total_count": 0,
                "total": 0,
                "mean": 0
            }, {
                "from": 1048576,
                "to": 2097152,
                "count": 0,
                "total_count": 0,
                "total": 0,
                "mean": 0
            }, {
                "from": 2097152,
                "to": 5242880,
                "count": 0,
                "total_count": 0,
                "total": 0,
                "mean": 0
            }, {
                "from": 5242880,
                "to": 10485760,
                "count": 0,
                "total_count": 0,
                "total": 0,
                "mean": 0
            }, {
                "from": 10485760,
                "count": 0,
                "total_count": 0,
                "total": 0,
                "mean": 0
            }]
        }
```

```
    },
    "total": 17
}
```

Query Parameters

- **query** – search text
- **onlyRoot** – only root items
- **totalSizeFrom** – total size from
- **totalSizeTo** – total size to
- **sort** – field to sort on
- **sort_direction** – sort direction, either `asc` (ascending) or `desc` (descending)
- **limit** – number of information object components to return
- **skip** – number of information object components to skip (an offset in other words)

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object AIPs

The information objectsi AIPs REST API endpoint allows information object AIPs data to be browsed.

GET /api/informationobjects/<id>/aips
Summary of information object AIP data.

Example request:

```
GET /api/informationobjects/508/aips HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "results": null,
    "facets": null,
    "total": null,
    "overview": {
        "unclassified": {
            "count": 0
        },
        "total": {
            "size": 0,
            "count": 0
        }
    }
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object Associate

The information object associate REST API endpoint allows the creation of associations between information objects.

POST /api/informationobjects/528/associate<id>

Create an association between two information objects.

Example request:

```
POST /api/informationobjects/528/associate HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "id": 653,
    "source_id": 508,
    "target_id": 506,
    "type_id": 176
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Information Object Assocations

The information object associations REST API endpoint provides data about a specific information object assocation and allows updating/deletion of assocations.

GET /api/informationobjects/association/<id>

Returns information object assocation data.

Example request:

```
GET /api/informationobjects/assocation/653 HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "id": 653,
    "subject": {
        "id": 508,
        "title": "UXnfY6wiU4D6VNATtiNx"
    },
    "object": {
        "id": 506,
        "title": "bYyaVdquKHifXa85nEQb"
    },
    "type": {
        "id": 176,
        "name": "Related material descriptions"
    }
}
```

Status Codes

- 200 OK – no error

PUT /api/informationobjects/assocation/1028

Update information object assocation data.

Example request:

```
  PUT /api/informationobjects/assocation/1028 HTTP/1.1
  Host: example.com

{
    "type_id": 176,
    "description": "These are linked"
}
```

Example response:

```
HTTP/1.1 200 OK
Content-Type: application/json
```

Status Codes

- 204 No Content – no content

DELETE /api/informationobjects/assocation/1028

Delete information object association data.

Example request:

```
DELETE /api/informationobjects/assocation/1028 HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

null
```

Status Codes

- 204 No Content – no content

Back to API documentation index

Back to API documentation index

2.1.6 Summary REST API

The summary REST API endpoints provide statistical data.

Ingestion Summary

The ingestion summary REST API endpoint lists how many of each type of AIP were ingested.

GET /api/summary/ingestion

Statistics about ingestions.

Example request:

```
GET /api/summary/ingestion HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "results": [{
        "total": 53,
        "type": "Artwork"
    }, {
        "total": 17,
        "type": "Supporting technology"
    }]
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Artwork by Date

The artwork by date summary REST API endpoint provides counts (and running totals) of how many artwork records were ingested, by month. Counts are also provided by collection date.

GET /api/summary/artworkbydate

Monthly counts of ingested artwork.

Example request:

```
GET /api/summary/artworkbydate HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "results": {
        "creation": [{
            "count": 53,
            "total": 53,
            "year": "2014",
            "month": "05"
        }],
        "collection": []
    }
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Storage by Media Category

The storage by media category summary REST API endpoint lists how much storage is used per MIME type.

GET /api/summary/storagebymediacategory

Storage used per MIME type.

Example request:

```
GET /api/summary/storagebymediacategory HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "results": [{
        "total": 3081552,
        "media_type": "audio\/mpeg"
    }, {
        "total": 12356184,
        "media_type": "video\/mpeg"
    }, {
        "total": 51554,
        "media_type": "image\/png"
    }, {
        "total": 6071934,
        "media_type": "video\/mp4"
    }]
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Storage by Codec

The storage by codec summary REST API endpoint provides stats about how much storage each codec is using.

GET /api/summary/storagebycodec

Statistics about file storage used by various codecs.

Example request:

```
GET /api/summary/storagebycodec HTTP/1.1
Host: example.com
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: application/json

{
    "results": [{
        "total": 10180006,
        "codec": "WAVE"
    }, {
        "total": 10180006,
        "codec": "QUICKTIME"
    }, {
        "total": 10180006,
        "codec": "MPEG VIDEO"
    }, {
        "total": 10180006,
        "codec": "MPEG AUDIO"
    }, {
        "total": 10180006,
        "codec": "AIFF"
    }, {
        "total": 10180006,
        "codec": "JPEG"
    }, {
        "total": 10180006,
        "codec": "MPEG-1V"
    }, {
        "total": 10180006,
        "codec": "PCM"
    }, {
        "total": 10180006,
        "codec": "MPA1L3"
    }]
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Artwork by Department

TODO: example endpoint

The artwork by department summary REST API endpoint provides stats about how many artworks are associated with each department.

GET /api/summary/departmentartworkcount
TO DO.

Example request:

```
GET /api/summary/departmentartworkcount HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: text/javascript

{
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Media File Size by Collection Year

TODO: example endpoint

The media file size by collection year REST API endpoint provides stats about TODO.

GET /api/summary/mediafilesizebycollectionyear
TODO.

Example request:

```
GET /api/summary/mediafilesizebycollectionyear HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: text/javascript

{
    "results": []
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

Back to API documentation index

2.1.7 Taxonomy REST API

The taxonomy REST API endpoint provides data about terms that exist in the system.

GET /api/taxonomies/<id>
List of taxonomy terms.

Example request:

```
GET /api/taxonomies/31 HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
  HTTP/1.1 200 OK
  Content-Type: text/javascript

{
    "terms": [{
        "id": 218,
        "name": "Full"
    }, {
        "id": 220,
        "name": "Minimal"
    }, {
        "id": 219,
        "name": "Partial"
    }]
}
```

Status Codes

- 200 OK – no error

Back to API documentation index

2.1.8 TMS Data REST API

Documentation goes here.

Back to API documentation index

2.1.9 User Authentication REST API

The users REST API endpoint allows a user to authenticate and returns data about the user.

GET /api/users
User information.

Example request:

```
GET /api/users HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
    HTTP/1.1 200 OK
    Content-Type: text/javascript

  {
  }
```

Status Codes

- 200 OK – no error

Back to API documentation index

Return to Binder documentation home

Binder is an open source digital repository management application designed to meet the needs and complex digital preservation requirements of cultural heritage institutions. Binder was created by Artefactual Systems and the Museum of Modern Art.

Binder aims to facilitate digital collections care, management, and preservation for time-based media and born-digital works, and is built from integrating functionality of the Archivematica and AtoM projects.

Important: Binder is still under development. Please see the *Current project status* page for more information.

Related resources

MoMA has created a screencast that offers an overview of Binder's functionality, which can be seen on YouTube:

- https://youtu.be/TelwvLkt-84

A presentation on Binder's functionality, given at AMIA 2014, can also be viewed online (note that Binder was initially called the DRMC):

- https://youtu.be/HPebm5nh83o

Slides from a presentation at the Code4LibBC 2014 conference can also be viewed at:

- http://www.slideshare.net/accesstomemory/introducing-the-drmc

See also:

- Binder on Github
- Binder User Forum

CHAPTER 3

User manual

User guide to creating, editing, importing, and managing content content in Binder via the user interface.

3.1 Overview

Find out what Binder is, what it can do, how it's being used, and how we hope to see the project evolve.

- *What is Binder?*
- *Technical overview*
- *Current project status*
- *Project goals: Binder's long-term vision*
- *Binder at MoMA: an example use case*

CHAPTER 4

Administration manual

Instructions for installing, upgrading, maintaining, and more; includes administering the app via the command-line interface.

4.1 API reference

Documentation of the avaiable REST endpoints used by Binder's HTTP API.

- *API reference*

/api