Preservica
Active digital preservation

OPEX: Instructions to select parent folder

Editions

This option is now available for all users of v6.3 and above from Essentials through to Enterprise Perform and for on-premise users.

Preservica Versions

This document is valid from v6.3.

Introduction

This note provides advice on using OPEX and the SourceID or Title to insert content into the required location in Preservica. Four use cases are included showing how this would be achieved.

Use Cases

At each ingest, the user needs their content which has been ingested using OPEX Incremental into different folders across their collection. If the user was using the PUT tool, then they would select that folder using Folder Options and identify an existing Folder.

Starting with an OPEX which consists of a Directory containing some files, here are four use cases:

1. the OPEX needs to be ingested **under** a folder at Root called *Pictures*
2. the OPEX needs to be ingested **under** a folder called *Films 1940s* which is **under** a folder at Root called *Films*
3. files in the OPEX need to be **added** to the folder called *Pictures*
4. the OPEX needs to be ingested **at Root** as a folder called *Books* which does not yet exist

SourceID

The SourceID (note capitalisation) is an element of the Transfer section of the .opex file e.g.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opex:OPEXMetadata xmlns:opex="http://www.openpreservationexchange.org/opex/v1.0">
  <opex:Transfer>
    <opex:SourceID>Pictures</opex:SourceID>
  </opex:Transfer>
</opex:OPEXMetadata>
```

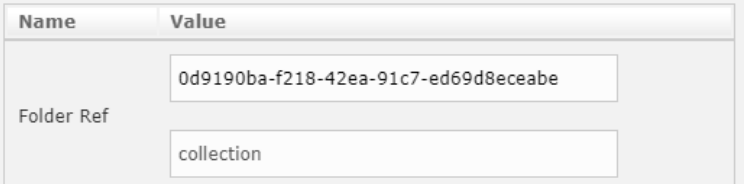
The SourceID is used to match Folders in the Ingest Package e.g. OPEX with existing Folders in Preservica with the same SourceID. SourceIDs should be unique. They derive from the idea that if you are copying a Folder and File structure from an external system then there may have been an idea in that “Source” that should be maintained. Additionally, if the export from the external system

is completed incrementally e.g. when the material reaches the date for permanent archival storage then the latest ingest will be merged into the existing structure using the SourceID.

If a SourceID is included in the .opex for a folder and the folder does not match with an existing folder based on SourceID, then the new folder created will be given the SourceID in the .opex so it is available for future ingests.

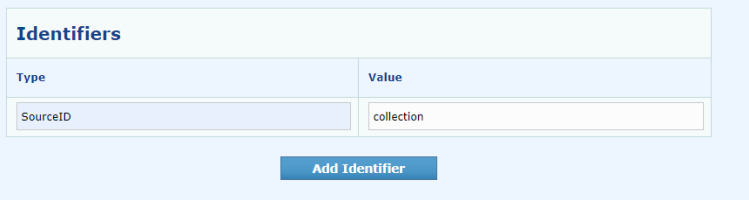
Set-up

In the Opex Incremental Workflow, always set up a default location for Ingest by updating the Folder Ref with a UUID of an existing Folder. This location is used when an alternate location is not given in the .opex metadata.



If OPEX and SourceIDs are being used with Folders that were ingested without a SourceID, the existing relevant folders need to be given a SourceID.

SourceIDs can be added from the Properties page using edit and Add Identifier.



The entity API can be used to manage Identifiers (Get, Post, Put and Delete) see: *https://demo.preservica.com/api/entity/documentation.html#/%2Fstructural-objects*

Care should be taken in determining SourceIDs so that they are unique.

As a minimum SourceIDs should be unique locally but if this is the case then the ingest process will rely on matching the full hierarchy above the ingest location. Additionally, consideration should be given to the possibility of re-ordering the collection and whether uniqueness can be maintained.

Use Case 1: the OPEX needs to be ingested **under** a folder at Root called *Pictures*

- Pictures folder already exists in Preservica at Root level.
- A SourceID needs to be present on the Pictures Folder at Root.
    - We assume in this example the SourceID is Pictures.

The OPEX to be ingested is structured:

Pictures
    Materials
    Pictures.opex
        File1.ext
        File2.ext
        Etc…
        Materials.opex

And placed inside a Container folder which is uniquely named and will not be ingested.

Pictures.opex:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opex:OPEXMetadata xmlns:opex="http://www.openpreservationexchange.org/opex/v1.0">
  <opex:Transfer>
    <opex:SourceID>Pictures</opex:SourceID>
  </opex:Transfer>
</opex:OPEXMetadata>
```

Materials.opex:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opex:OPEXMetadata xmlns:opex="http://www.openpreservationexchange.org/opex/v1.0">
  <opex:Transfer>
    <opex:SourceID>Materials</opex:SourceID>
  </opex:Transfer>
</opex:OPEXMetadata>
```

On ingest:

- The Pictures Directory[1] in the OPEX will use the SourceID to match to the Pictures Folder in Preservica.
    - It will not create a new Pictures folder.
    - The Monitor will report:
      Matched directory Pictures to existing SO [UUID] via Source ID Pictures
- The Materials Directory in the OPEX will be created on ingest because there is no folder under Pictures with the SourceID of Materials in Preservica.
    - The Monitor will report
      Created new SO [UUID]
    - The new folder will be given the SourceID Materials
- The Files will be ingested into the Materials folder

Note: It is not mandatory to include the Materials.opex file but if it is not included then (a) matching will only be possible if the full hierarchy is used in future ingests and (b) the folder name is unique.

- Folder matching defaults to using the Title locally when a SourceID is not present.

[1] Directory is used to describe a Folder in your network drive or the location that the OPEX is created. This is to differentiate from Folders in Preservica.

Use Case 2: the OPEX needs to be ingested **under** a folder called *Films 1940s* which is **under** a folder at Root called *Films*

Option 1 for use where SourceIDs are unique.

- Films 1940s folder already exists in Preservica under Films.
- A SourceID needs to be present on the Films 1940s Folder.
    - We assume in this example the SourceID is Films 1940s.
- It is not mandatory for the Films folder to have a SourceID

The OPEX to be ingested is structured:

Films 1940s
    Casablanca
    Films 1940s.opex
        File1.ext
        File2.ext
        Etc…
        Casablanca.opex

And placed inside a Container folder which is uniquely named and will not be ingested.

Films 1940s.opex:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opex:OPEXMetadata xmlns:opex="http://www.openpreservationexchange.org/opex/v1.0">
  <opex:Transfer>
    <opex:SourceID>Films 1940s</opex:SourceID>
  </opex:Transfer>
</opex:OPEXMetadata>
```

Casablanca.opex:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opex:OPEXMetadata xmlns:opex="http://www.openpreservationexchange.org/opex/v1.0">
  <opex:Transfer>
    <opex:SourceID>Casablanca</opex:SourceID>
  </opex:Transfer>
</opex:OPEXMetadata>
```

On ingest:

- The Films 1940s Directory in the OPEX will use the SourceID to match to the Films 1940s Folder in Preservica.
    - It will not create a new Films 1940s folder.
    - The Monitor will report:
      Matched directory Films 1940s to existing SO [UUID] via Source ID Films 1940s
- The Casablanca Directory in the OPEX will be created on ingest because there is no folder under Films 1940s with the SourceID of Casablanca in Preservica.
    - The Monitor will report
      Created new SO [UUID]
    - The new folder will be given the SourceID Casablanca
- The Files will be ingested into the Casablanca folder

Option 2 for use where SourceIDs are NOT unique.

- Films 1940s folder already exists in Preservica under Films.
- A SourceID needs to be present on the Films 1940s AND the Films Folder.
    - We assume in this example the SourceIDs are Films 1940s and Films

The OPEX to be ingested is structured:

Films
- Films 1940s
- *Films.opex*
  - Casablanca
  - *Films 1940s.opex*
    - File1.ext
    - File2.ext
    - Etc…
    - Casablanca.opex

And placed inside a Container folder which is uniquely named and will not be ingested.

Films.opex:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opex:OPEXMetadata xmlns:opex="http://www.openpreservationexchange.org/opex/v1.0">
  <opex:Transfer>
    <opex:SourceID>Films</opex:SourceID>
  </opex:Transfer>
</opex:OPEXMetadata>
```

Films 1940s.opex:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opex:OPEXMetadata xmlns:opex="http://www.openpreservationexchange.org/opex/v1.0">
  <opex:Transfer>
    <opex:SourceID>Films 1940s</opex:SourceID>
  </opex:Transfer>
</opex:OPEXMetadata>
```

Casablanca.opex:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opex:OPEXMetadata xmlns:opex="http://www.openpreservationexchange.org/opex/v1.0">
  <opex:Transfer>
    <opex:SourceID>Casablanca</opex:SourceID>
  </opex:Transfer>
</opex:OPEXMetadata>
```

On ingest:

- The Films Directory in the OPEX will use the SourceID to match to the Films Folder in Preservica.
  - It will not create a new Films folder.
  - The Monitor will report:
    Matched directory Films to existing SO [UUID] via Source ID Films
- The Films 1940s Directory in the OPEX will use the SourceID to match to the Films 1940s Folder in Preservica.
  - It will not create a new Films 1940s folder.
  - The Monitor will report:
    Matched directory Films 1940s to existing SO [UUID] via Source ID Films 1940s
- The Casablanca Directory in the OPEX will be created on ingest because there is no folder under Films 1940s with the SourceID of Casablanca in Preservica.
  - The Monitor will report
    Created new SO [UUID]
  - The new folder will be given the SourceID Casablanca
- The Files will be ingested into the Casablanca folder

Use Case 3: files in the OPEX need to be added to the folder called *Pictures*

Assuming the SourceIDs are unique this method below can be used, if they are not then the full folder hierarchy up to Root with accompanying .opex files must be added.

- Pictures folder already exists in Preservica.
- A SourceID needs to be present on the Pictures Folder.
    - We assume in this example the SourceID is Pictures.

The OPEX to be ingested is structured:

Pictures
    File1.ext
    File2.ext
    Etc…
    PIctures.opex

And placed inside a Container folder which is uniquely named and will not be ingested.

Pictures.opex:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opex:OPEXMetadata xmlns:opex="http://www.openpreservationexchange.org/opex/v1.0">
  <opex:Transfer>
    <opex:SourceID>Pictures</opex:SourceID>
  </opex:Transfer>
</opex:OPEXMetadata>
```

On ingest:

- The Pictures Directory in the OPEX will use the SourceID to match to the Pictures Folder in Preservica.
    - It will not create a new Pictures folder.
    - The Monitor will report:
      Matched directory Pictures to existing SO [UUID] via Source ID Pictures
- The Files will be ingested into the Pictures folder

Use Case 4: the OPEX needs to be ingested **at Root** as a folder called *Books* which does not yet exist.

Folders cannot be created at Root level using OPEX. Without SourceIDs, the ingest will fall back to the Folder Ref in the OPEX Incremental Ingest workflow and OPEX Folders will be placed UNDER the Folder listed in the FolderRef.

To deliver this use case if a Books Folder is created at Root level prior to ingest and the method describe in Use Case 3 is used for adding files to the Books Folders then the end result can be achieved.