Eluvio Content Fabric Media Ingest Guide (PREVIEW)

THIS DOCUMENTATION IS A WORK IN PROGRESS, SOME TOPICS ARE CURRENTLY STUBS, SOME ARE PENDING REVISION

Overview / Basic concepts

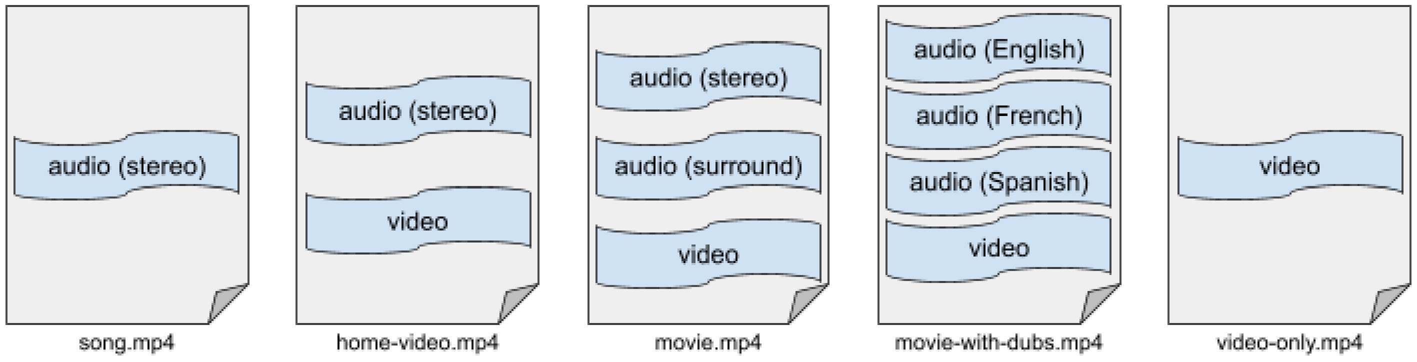
Media Files and Streams

If you are already familiar with the structure of media files, skip ahead to the Masters and Mezzanines section.

Ingesting on-demand (as opposed to live) content begins with 1 or more **Media Files**.

A Media File contains 1 or more **Streams**. A Stream is one 'track' of media (audio, video, subtitles, or other data).

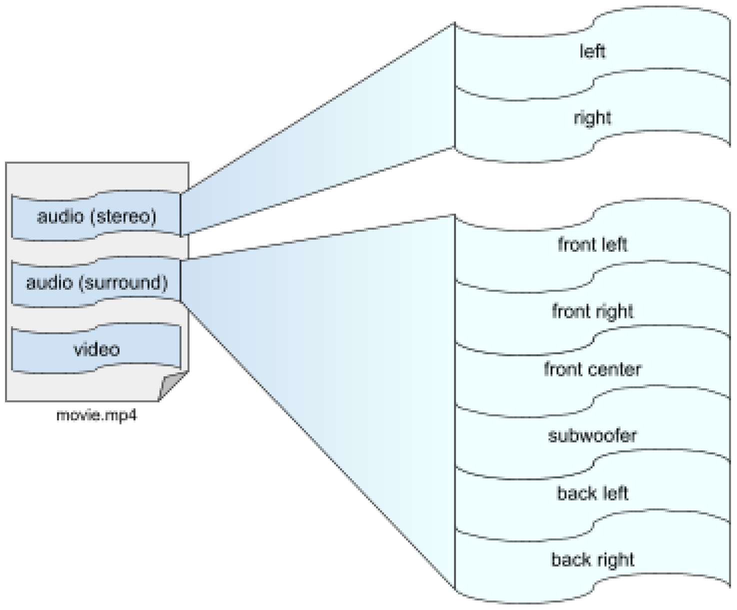
Most Media Files contain at most 1 video stream but may have more than 1 audio stream:



In the simplest case, an ingest starts with a single Media File that has 1 stereo audio stream and 1 video stream. In these cases there is no manual input needed to choose which file to use or which stream(s) to include.

Often, however, things are more complex - there may be multiple files that make up a title, or multiple streams to choose from. More on this in the Variants section.

Audio Streams are further subdivided into **Channels**. A channel is generally the audio to be sent to a single speaker:

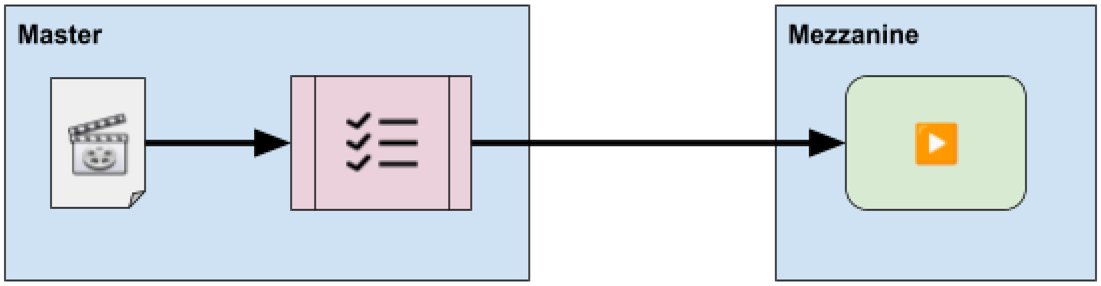


In summary, a **Media File** contains 1 or more **Streams**. An audio **Stream** contains 1 or more **Channels**.

Masters and Mezzanines

In the Eluvio Content Fabric, ingesting media for streaming involves two kinds of objects, **Masters** and **Mezzanines**.

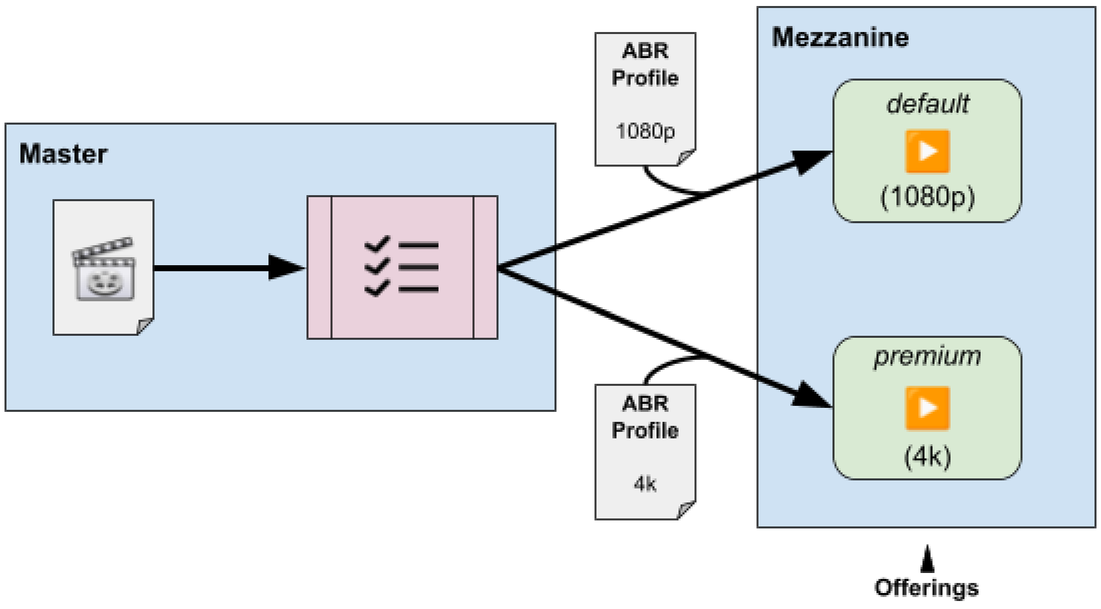
A **Master** organizes your original source files . A **Mezzanine** holds a version of your content optimized for low latency streaming, and is generated from your **Master** file(s) based on a stream list that is configured in the master:



Master	**Mezzanine**
Accessible only to content owners / admins	Accessible to (specified) users or the public
Not directly viewable from Content Fabric (yet)	Directly viewable from the Content Fabric, and optimized for low latency streaming
Often very high bitrate/resolution	Usually lower bitrate and/or resolution than Master
Often encoded with professional codecs (Prores / JPEG2000 / PCM-raw Audio etc.)	Encoded with h264 or h265 video and AAC audio
Media optionally stored outside of Content Fabric in S3	(Transcoded) media is stored in the Content Fabric

Offerings and ABR Profiles

The viewable / streamable item in the **Mezzanine** is called an **Offering**. It makes your media playable in various resolutions / bit rates and playout formats. These playout options are set using an **ABR Profile**. It is possible for a mezzanine to contain multiple **Offerings** ingested with different settings, for example to create a "premium" **Offering** with higher quality playback:



The name given to the **Offering** is called the ***offering key***. This becomes part of the URL used for playout. Playout access control can be configured on a per-offering basis.

If no offering key is specified during ingest then `"default"` will be used.

Every ingest must use an **ABR Profile**. It configures the following:

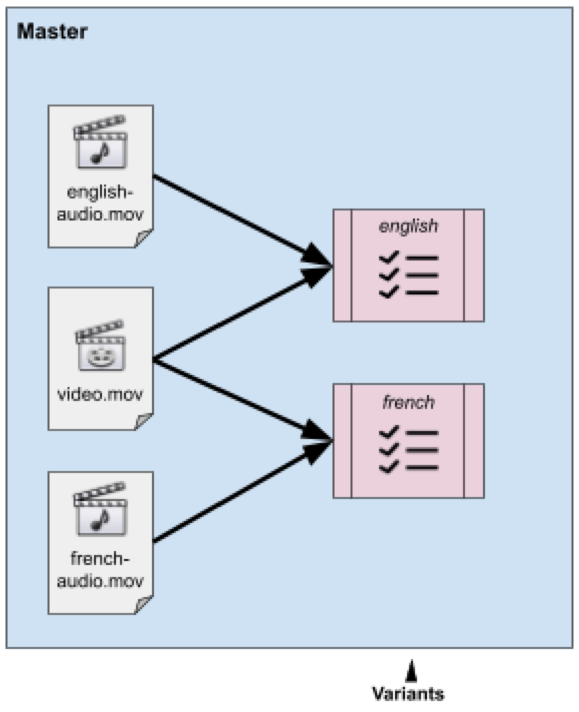
- Top resolution / bitrate
- Additional (lower) resolutions / bitrates to make available
- Playout format(s)
- DRM type(s)
- Watermarking

After ingest, an **Offering** can be edited to change all of the above settings ***except*** "Top resolution / bitrate" and "Storage encryption". A change to either of those two settings requires re-transcoding and recreating the **Offering**.

Variants

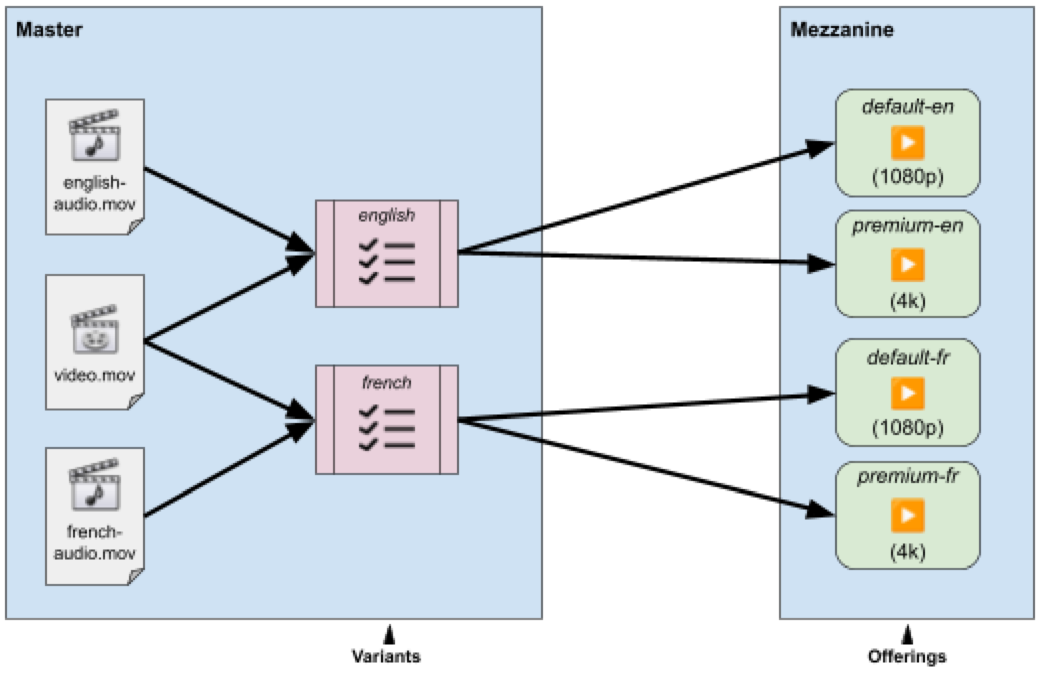
The list in the **Master** that controls what streams to include (and from which files) is called a **Variant**.

It is possible for a **Master** to define multiple **Variants**, e.g. for internationalization:



The name given to the **Variant** is called the ***variant key***. It is an internal reference, independent of the ***offering key***. (If no variant key is specified, `"default"` will be used)

Each **Offering** makes one **Variant** playable in the formats specified by an **ABR Profile**. A **Variant** may have more than one **Offering** created from it.

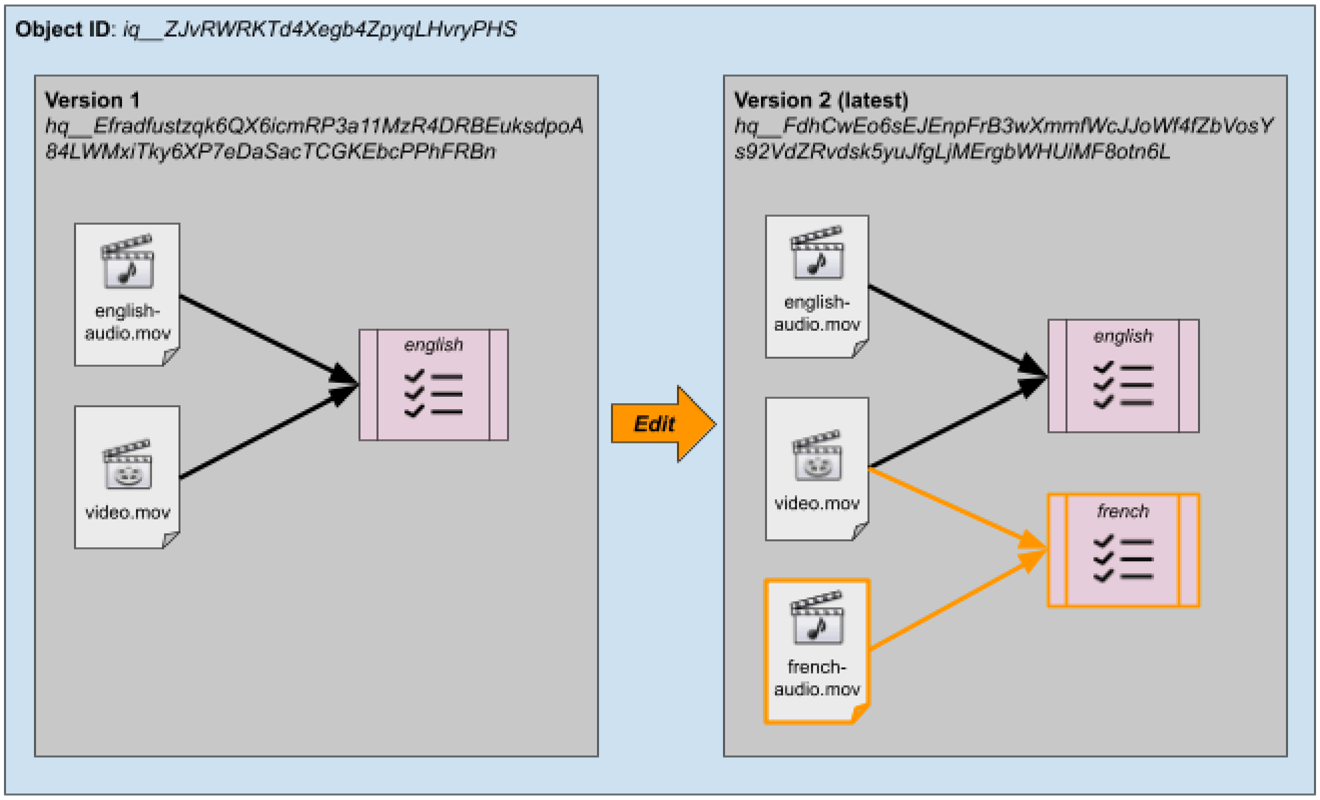


Objects and Versions

The basic unit of organization in the Content Fabric is called an **Object**. It is a container for holding information such as files and metadata.

Every **Object** has a unique ***Object ID*** starting with `iq__`, e.g. ***`iq__4JhZtSENmDu7hTq91LEufzDnDpj3`***.

Each time you save changes to an **Object** the Content Fabric will create a new **Version** of it. Every **Version** has a ***Version Hash*** starting with `hq__` which identifies it and also acts as a signature to ensure that its contents have not been tampered with. (The ***Object ID*** remains the same when you edit an ***Object***.)


Object ID: iq__ZJvRWRKTd4Xegb4ZpyqLHvryPHS
Version 1
hq__Efradfustzqk6QX6icmRP3a11MzR4DRBEuksdpoA84LWMxiTky6XP7eDaSacTCGKEbcPPhFRBn
english-audio.mov
video.mov
english
Edit
Version 2 (latest)
hq__FdhCwEo6sEJEnpFrB3wXmmfWcJJoWf4fZbVosYs92VdZRvdsk5yuJfgLjMErgbWHUiMF8otn6L
english-audio.mov
video.mov
french-audio.mov
english
french

Past **Versions** of an **Object** are kept, and you can view them in the Fabric Browser by clicking the `Show Previous Versions` button.

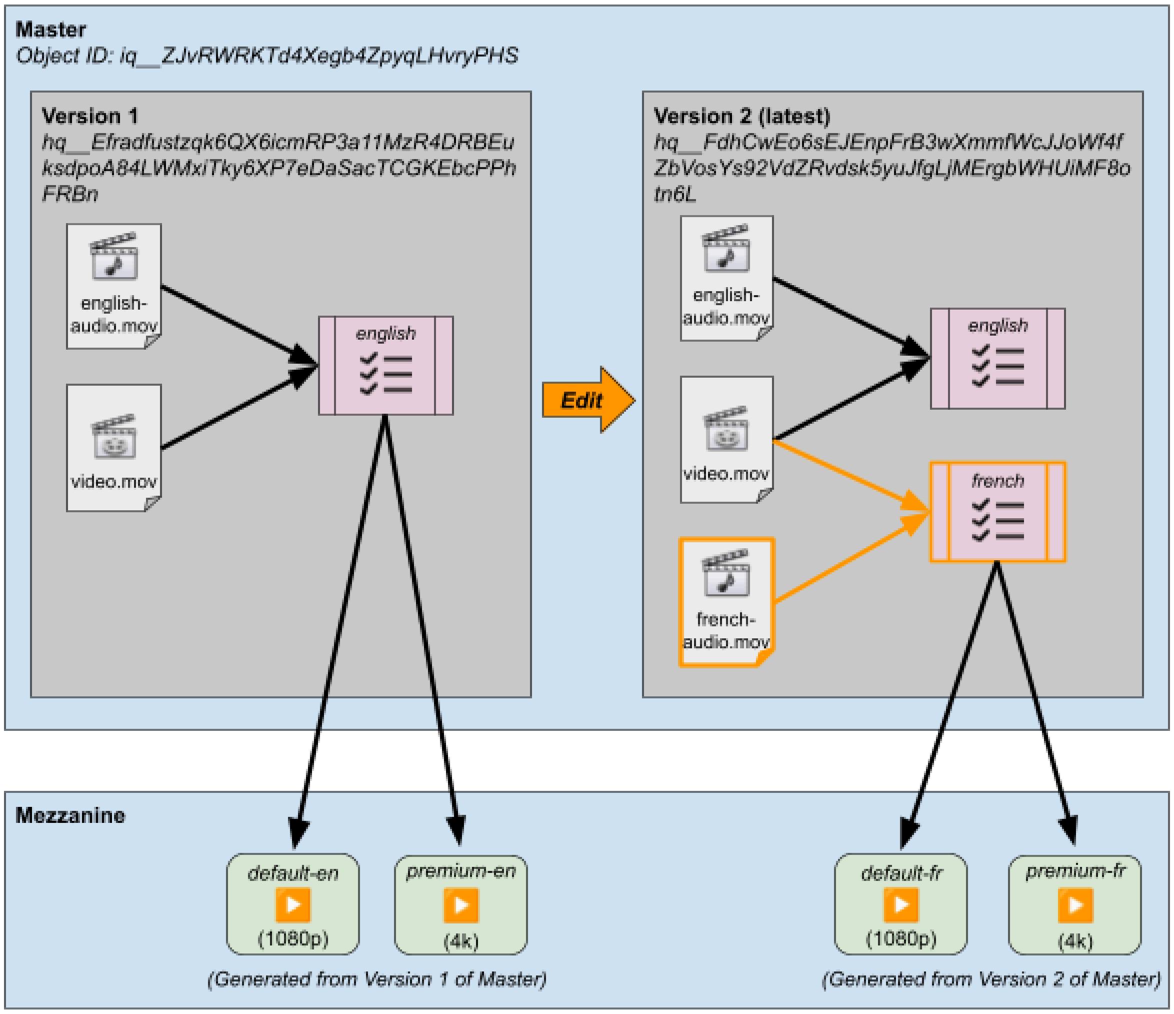
Master Version Hashes

When generating an **Offering**, you must specify what **Version** of the **Master** you wish to base it on. (Usually you will want to use the latest **Version** of the **Master**). This is done by specifying the ***Master Version Hash*** that you want to base your **Offering** on.

Version Hashes begin with `hq__`.

Updating the **Master** does not update any **Offerings** which have already been created, and they will continue to be based on the older **Version** of the **Master**.

This is fine if you haven't made any changes that need to be copied to the old **Offering**. In the example below, the **Master** was edited to add a French audio source file and create a **Variant** called "french". This is not a change that would affect the previously generated **Offerings** ("default-en" and "premium-en"), so they do not need to be regenerated.


Master
Object ID: iq__ZJvRWRKTd4Xegb4ZpyqLHvryPHS
Version 1
hq__Efradfustzqk6QX6icmRP3a11MzR4DRBEuksdpoA84LWMxiTky6XP7eDaSacTCGKEbcPPhFRBn
english-audio.mov
video.mov
english
Edit
Version 2 (latest)
hq__FdhCwEo6sEJEnpFrB3wXmmfWcJJoWf4fZbVosYs92VdZRvdsk5yuJfgLjMErgbWHUiMF8otn6L
english-audio.mov
video.mov
french-audio.mov
english
french
Mezzanine
default-en (1080p)
premium-en (4k)
(Generated from Version 1 of Master)
default-fr (1080p)
premium-fr (4k)
(Generated from Version 2 of Master)

On the other hand, if you discovered a problem with the English audio track and updated the **Master** (either replacing the original file or adding a file with a new filename) then you would need to regenerate the "default-en" and "premium-en" **Offerings**, using a new ***Master Version Hash***.

Tenancies

To provide access to the Content Fabric, a **Tenancy** was set up for your organization. This involves creating private keys (i.e. accounts) and configuring various storage areas, permissions, and so forth. The name used when setting up your Tenancy will appear as a prefix in the names of various items such as **Libraries**, **Content Types**, and **Access Groups**. All of the output examples in this document use a **Tenancy Name** of `dev-tenant`.

Tenancies also control billing for the Eluvio Content Fabric.

Tenancies have Content Fabric IDs beginning with `iten`. While you will not usually need to refer to your Tenancy ID, you can view it by clicking on the **ELUV.IO** logo in the Fabric Browser, then clicking **Profile** and looking at the **Current Tenant** section.

Libraries

Content Fabric **Objects** are contained in **Libraries** (every Object belongs to one Library).

Libraries have IDs that start with `ilib`.

When your Tenancy was set up, the following four Libraries were created:

- `TENANCY_NAME - Properties` (web sites)
- `TENANCY_NAME - Reports` (usage reports)
- `TENANCY_NAME - Title Masters` (media master objects)
- `TENANCY_NAME - Title Mezzanines` (playable mezzanine objects)

It is possible to create additional Libraries if needed.

Content Types

Content Fabric **Objects** can be assigned a **Content Type** which adds certain capabilities. For example, generally Mezzanines are assigned a Content Type of "Title" during creation, which enables any Offerings it contains to be played in streaming format.

Content Types have IDs that start with `iq__` (the same as regular Object IDs)

When your Tenancy was set up, a variety of Content Types were created, but for ingesting media we will only need to work with two:

- `TENANCY_NAME - Title` (for Mezzanines)
- `TENANCY_NAME - Title Master` (for Masters)

Getting started

Prerequisites

You will need the following installed:

- git (type `git --version` to check if it is installed)
- node.js (at least version 14.x - type `node -v` to check version)

You will also need a Content Fabric Private Key - please check with the Eluvio rep who set up your tenancy.

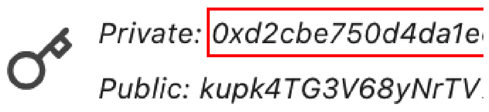
Open the Content Fabric Browser page

If you have been set up on our **demo** network, go to:
https://core.demov3.contentfabric.io/#/apps/Eluvio%20Fabric%20Browser/#/

If you have been set up on our **production** network, go to:
https://core.v3.contentfabric.io/#/apps/Eluvio%20Fabric%20Browser/#/

Find your Content Fabric private key and record it in a safe place

- Enter the password you chose when first setting up your account
- Click on the **ELUV.IO** logo at top left
- Click on **Profile**
- Click on the key icon in the middle of the page to reveal your keys
- Double-click on the item marked ***Private:*** and copy it



Private: 0xd2cbe750d4da1e
Public: kupk4TG3V68yNrTV

- **IMPORTANT:** Save in a safe place (a permanent file). Do not share this key. The Content Fabric is designed to be trustless - **we do not keep a copy of your key and cannot reset or recover it for you**. Until you save a copy of your key somewhere it only exists in your browser's local storage, which can get erased if you choose to wipe your browser history.

Download and set up elv-utils-js

1. If you do not have them already, install git and node.js / npm *(recommended minimum version of node.js is 16.x)*
2. Navigate to the directory where you would like to install **elv-utils-js** and run the following commands:

```
git clone https://github.com/eluv-io/elv-utils-js
cd elv-utils-js
npm install
```

At this point, you may see some warnings about package vulnerabilities. As you will only be running local command line scripts (rather than starting a web server that accepts input from the outside world) these are safe to disregard.

Determine your Fabric configuration URL

If you have been set up on the Demo network, the URL is https://demov3.net955210.contentfabric.io/config

If you have been set up on the Production network, the URL is https://main.net955305.contentfabric.io/config

Set environment variables

The `elv-utils-js` scripts use environment variables to supply certain information. These will need to be set whenever you start a new terminal session.

1. Configure the `FABRIC_CONFIG_URL` environment variable, e.g. for production network this would be:

```
export FABRIC_CONFIG_URL="https://main.net955305.contentfabric.io/config"
```

2. Configure the `PRIVATE_KEY` environment variable:

```
export PRIVATE_KEY=0x...  (your Content Fabric private key)
```

3. If your media files are hosted on AWS S3, also set the following environment variables:

```
export AWS_REGION=     (your AWS region)
export AWS_BUCKET=     (your AWS bucket name)
export AWS_KEY=AK...   (your AWS S3 key)
export AWS_SECRET=...  (your AWS S3 secret)
```

Look up your Content Type IDs and LibraryIDs

(Make sure you have set your environment variables first, see previous section)

1. Change your current directory to `elv-utils-js/utilities`:

```
cd elv-utils-js/utilities
```

2. Look up your Content Type IDs with `ListTypes.js`

```
node ListTypes.js
```

Sample output:

```
Get list of content types

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)

id                                name
iq__3j9xhdqvzkGvzk5cGSBsD4mk12TG dev-tenant - Channel
iq__3168aeRL4ve6YAE4HX1huEVW6E6T dev-tenant - Eluvio LIVE Drop Event Site
iq__MeiZ3F266sx1hk3QvccVsLhQKfp  dev-tenant - Eluvio LIVE Marketplace
iq__bdQBYaURdXCiHZNkH78B56wQ6eJ  dev-tenant - Eluvio LIVE Tenant
iq__4U84NCVvoVH51249dFDobj2ZfZrK dev-tenant - Live Stream
iq__2jh5w9vsnE16P2tbd2xNiazcYsbZ dev-tenant - NFT Collection
```

```
iq__3oGYU8DaVfjsu4GiyALzVvnBcYfU dev-tenant - NFT Template
iq__4WQZhWEPmRT8rq5DLSJxBMnRhBAk dev-tenant - Permissions
iq__8SLzhEyJWiJ41BPezhswG56MUwL  dev-tenant - Title
iq__TSDVBqGVstjJzhYBXuWTKE2AZaw  dev-tenant - Title Collection
iq__2tfLjovW8zMN9Yh6eLmwynX1Cbip dev-tenant - Title Master

Done.
```

3. Copy the IDs for the rows that end with `" - Title"` and `" - Title Master"`. The first is the Content Type ID for your **Mezzanines**, the second is the Content Type ID for your **Masters**.

4. Look up your Library IDs with `ListLibraries.js`

```
node ListLibraries.js --name
```

Sample output:

```
Get list of libraries

Getting list of libraries...
Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Found 4 lib(s)

libraryId                        name
ilib3MrFBvGaJeL5rCJeMg2KGPZsVecH dev-tenant - Reports
ilib2nLKiR5p2yiGqCNicszxQYyvu9W4 dev-tenant - Properties
ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3 dev-tenant - Title Masters
ilib29dvmbN91uyXRwcMX88CAs8q2zeT dev-tenant - Title Mezzanines

Done.
```

5. Copy the IDs for the rows that end with `" - Title Masters"` and `" - Title Mezzanines"`. *(Note that the order is the reverse of Content Types - the Master library is listed before the Mezzanine library)*

Look up the address of your 'Content Admins' Access Group

Run the `ListAccessGroups.js` script:

```
node ListAccessGroups.js
```

Sample output: *(you will see your own tenancy name instead of* ***`dev-tenant`****)*

```
List access groups visible to the currently configured private key

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)

address                                    name
0x8d8780cfa0970a064e247e4a7829f0106b38d7f7 dev-tenant Content Admins
0xde9742280f18724d7b6b2d8bc7f3d96b538bc265 dev-tenant Content Users
0xce2a975ee73a8091cc7d451ed7c74a1de2739617 dev-tenant Tenant Admins


Done.
```

Copy the address for your Content Admins group.

Your first ingest

To test that everything is configured properly, start with a file that is:

- Short (several minutes or less)
- Local (i.e. not on S3)
- Simple (1 video stream and 1 audio stream)

Make sure that the `FABRIC_CONFIG_URL` and `PRIVATE_KEY` environment variables are set, and that you are in the `elv-utils-js/utilities` directory.

At this point, you should have the following information collected for copying and pasting into commands:

- Content Type and Library ID for **Masters**
- Content Type and Library ID for **Mezzanines**
- Address of your **Content Admins Group**

For example, the values collected from the output samples in the previous section are shown below ***(your values will be different!)*** Note that Library IDs start with `ilib`, Content Type IDs start with `iq__`, and the Access Group address starts with `0x`:

```
Master content type  iq__2tfLjovW8zMN9Yh6eLmwynX1Cbip
Master library       ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Mez content type     iq__8SLzhEyJWiJ41BPezhswG56MUwL
Mez library          ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Content Admins group 0x8d8780cfa0970a064e247e4a7829f0106b38d7f7
```

In the commands that follow, substitute your own values for the items that start with a dollar sign. `$MASTER_TYPE` and `$MEZ_TYPE` are the Content Type IDs, `$MASTER_LIB` and `$MEZ_LIB` are the Library IDs, and `$ACCESS_GROUP_ADDRESS` is the "Content Admins" Access Group address. For example, the `MasterCreate.js` command might look like the following after substitution:

```
node MasterCreate.js \
  --libraryId ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3 \
  --type iq__2tfLjovW8zMN9Yh6eLmwynX1Cbip \
  --title "Test 1" \
  --files ~/Downloads/Video.mp4
```

1. Create the Master:

```
node MasterCreate.js \
  --libraryId $MASTER_LIB \
  --type $MASTER_TYPE \
  --title "$YOUR_TITLE" \
  --files $PATH_TO_YOUR_FILE
```

Sample output:

```
Create production master

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up content type: iq__2tfLjovW8zMN9Yh6eLmwynX1Cbip...
Video.mp4: 0.0%
Video.mp4: 100.0%

Production master object created:
  Object ID: iq__4JhZtSENmDu7hTq91LEufzDnDpj3
  Version Hash: hq__6siNFi29bLmTmDAoSJ22t1dbszC7j2Y9KU2YupEuk2HdDr2LuhE5BMmJN6m5sRzJdn7UDpqGDF


Done.
```

2. Copy the **Version Hash** output by the `MasterCreate.js` script. This will need to be substituted in for `$MASTER_VERSION_HASH` in the next command.
3. Create the **Mezzanine** Object (and wait for transcoding to finish)

```
node MezCreate.js \
  --libraryId $MEZ_LIB \
  --type $MEZ_TYPE \
  --title "$YOUR_TITLE" \
  --masterHash $MASTER_VERSION_HASH \
  --abrProfile ../example-files/abr_profile_no_drm_store_encrypted.json \
  --wait
```

Sample output:

```
Create Mezzanine offering 'default' for master
       hq__6siNFi29bLmTmDAoSJ22t1dbszC7j2Y9KU2YupEuk2HdDr2LuhE5BMmJN6m5sRzJdn7UDpqGDF

Reading file elv-utils-js/example-files/abr_profile_no_drm_store_encrypted.json...
Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up content type: iq__8SLzhEyJWiJ41BPezhswG56MUwL...
Creating new mezzanine object...
Starting Mezzanine Job(s)

Library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Object ID: iq__i4YaCPa6NTt5747C5XWeoWEPC3K
Offering: default
Write Token:
       tqw__HSRwEN24A4Di7L3fUDX3WdvgE8arN7RyyVdAbY8mG4i9wKhJbc7SbuUVm9o7Yp7311az6wRGpvmwdYoLMCf
Write Node: http://localhost:8008/

Progress:
run_state: running
estimated time left: unknown (not enough progress yet)
run_state: finished

ABR mezzanine object created:
  Object ID: iq__i4YaCPa6NTt5747C5XWeoWEPC3K
  Version Hash: hq__LmVFMGjnvTRR61J4iPMajq9o4qEt3QnfopncBCUfVx9ggWPzfG9KayeCf5CMHdZj4XnjxiDcy7

Waiting for publishing to finish and new object version to become available...
  new version not available yet, waiting 15 seconds...
New object version now available

Done.
```

4. Copy the **Object ID** in the output. This will need to be substituted in for `$MEZ_OBJECT_ID` in the next command.

5. Grant **`manage`** permission for the Mezzanine Object to the your **Content Admins** group:

```
node ObjectAddGroupPerm.js \
  --objectId $MEZ_OBJECT_ID \
  --groupAddress $ACCESS_GROUP_ADDRESS \
  --permission manage
```

   Sample output:

```
Add 'manage' permission to iq__ksXLW12ZAAg7PcLS4gKLUPPRUin for group
       0x8d8780cfa0970a064e247e4a7829f0106b38d7f7

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Adding permission: manage...

Done.
```

The **Object ID** you copied in step 4 is the ID of your Mezzanine, and you can use the ID to jump directly to your mezzanine in the Fabric Browser:

- Paste the Object ID into the **`Find content by ID`** box at top right of Fabric Browser
- Press Enter or click the **`Search`** button

You can also find your Mezzanine by clicking **`Content`** on the left sidebar and then on the row for your **`"Title Mezzanines"`** library.

Once you are on the page for your Mezzanine object, click on the **`Display`** tab to play it (player will start with sound muted).

(OPTIONAL) There are two additional steps that are common for most use cases:

6. Also grant **`manage`** permission for the **Master** Object to your **Content Admins** group (you will need to copy the **Object ID** from output of step 1, "Create the Master", and substitute it for ***`$MASTER_OBJECT_ID`***)

```
node ObjectAddGroupPerm.js \
  --objectId $MASTER_OBJECT_ID \
  --groupAddress $ACCESS_GROUP_ADDRESS \
  --permission manage
```

7. Set the **Object Permission** on Mezzanine to **`viewable`**:

```
node ObjectAddGroupPerm.js \
  --objectId $MEZ_OBJECT_ID \
  --groupAddress $ACCESS_GROUP_ADDRESS \
  --permission manage
```

Sample output:

```
Set permission on iq__ksXLW12ZAAg7PcLS4gKLUPPRUin to viewable

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Previous version hash:
        hq__ErxfDQX4tExvstzb8URyx9WG2BYBrWkAWB66kvzH6Hw3pkQKZzN8PmL6AGAftLyktCcataFxSW
New version hash: hq__FH1ZgDNyqr2Hj9eFRNNMvSdKfSpU3x5dpPufo2xWbnDKqiXrp9XFV7jDfgoFaXTwTt7aDgVh1t

Done.
```

Next steps / Advanced topics

Storage Encryption

NOTE: THIS SECTION STILL NEEDS REVISION.

By default, items in the Content Fabric are stored in **encrypted** form, and a private key is required to authorize access and decrypt. For some use cases, however, you may wish to make something available for playout to the general public without any authorization at all (e.g. a movie trailer). In these cases the data must be stored **without encryption** (aka "clear").

If you wish to create an Offering that allows playout without authorization you must use an ABR Profile that specifies `"store_clear": true`

In the `elv-utils-js/example-files` directory, these are the files with names ending in "`_store_clear.json`".

Note that DRM is not available on Offerings ingested using clear storage. If you change your mind and wish to add DRM later, you will need to re-ingest using an ABR Profile that specifies `"store_clear": false`

Access Groups

NOTE: THIS SECTION STILL NEEDS REVISION.

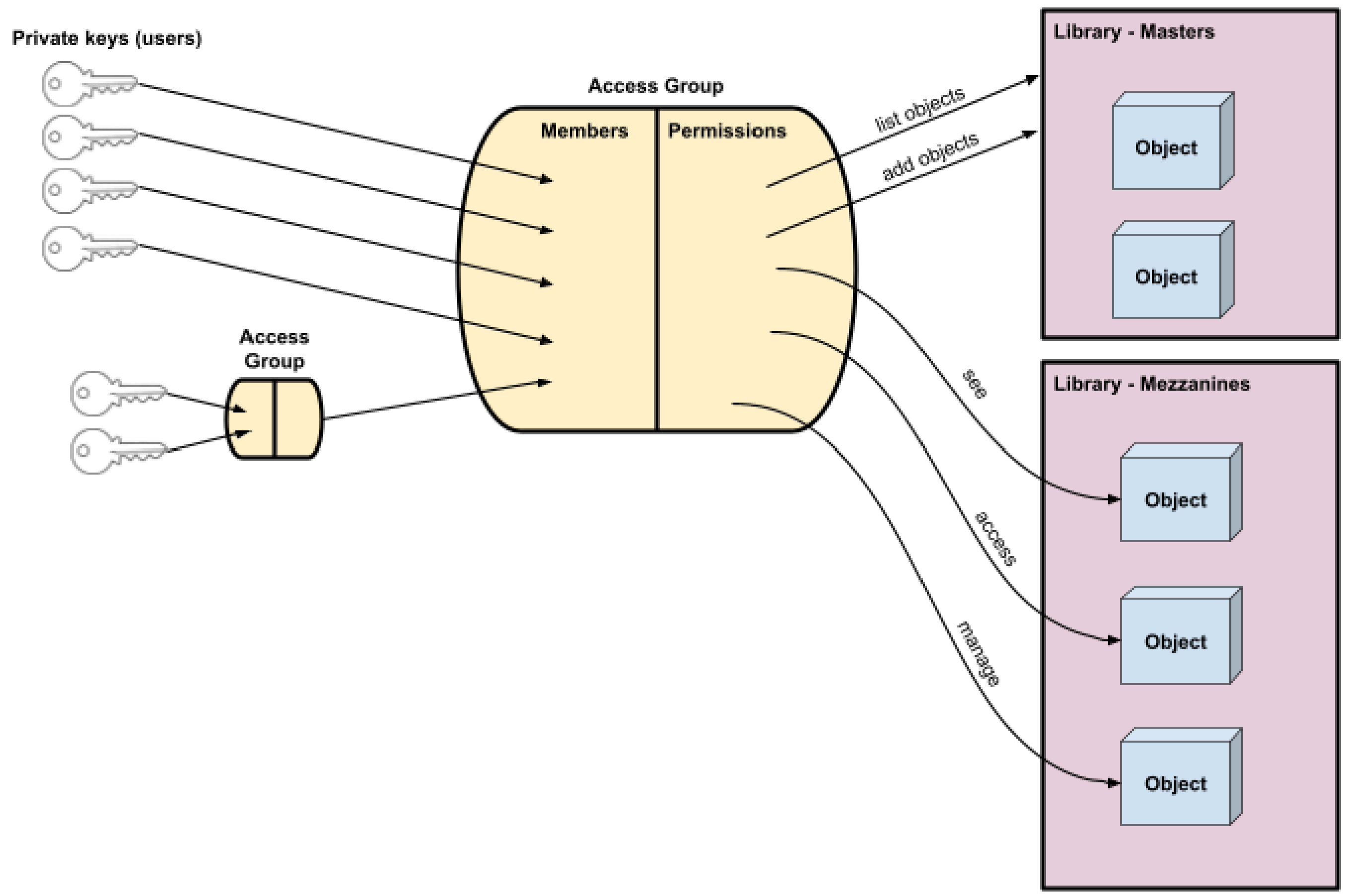
One of the ways that the Content Fabric organizes and controls access to Objects is via **Access Groups**. These are groups that can be granted permissions to Libraries and Objects, and whose members are either individual users (i.e. private keys) or other Access Groups.

There are 2 different kinds of permissions for Libraries:

- `list objects` - see the list of objects in the library (and each object's name and other public metadata)
- `add objects` - create and add new objects to the library

There are 3 different kinds of permissions for Objects.:

- `see` - See only public metadata for the object
- `access` - See all details of the object, public and private
- `manage` - See all details of the object, public and private, as well as edit the object



When your tenancy was set up, the following three Access Groups were created:

- `*TENANCY_NAME* Content Admins` (can create new objects in Libraries)
- `*TENANCY_NAME* Content Users` (cannot create new objects in Libraries)
- `*TENANCY_NAME* Tenant Admins` ()

Note that having "list objects" permission for a Library automatically grants "see" access to all Objects within.

Object Permission / Visibility

NOTE: THIS SECTION STILL NEEDS REVISION.

Each Object in the Content Fabric has a base permission level setting. This **Object Permission** setting can be misleading however, as **it can be overridden by granting Access Groups permissions on an Object.**

Just because an object is set to "Owner Only" does not necessarily mean that no one else can read or edit the object. For example, after creating a Mezzanine, the owner could then give the `Content Admins` group permission to `manage` the Object. This will allow any member of that Access Group the ability to edit the Mezzanine, regardless of the Object Permission setting.

Level	**Permissions**
Owner Only	Only the owner of the object (that is, the creator) can access the object or change its permissions. No one but the owner can read the object. ***(but see above!)***
Editable	Members of Access Groups that have **"manage"** permission on the object have full access to it including the ability to write to the object and to change its permissions.
Viewable	Members of Access Groups that have **"access"** permission on the object have full **read-only** access to it. Members of Access Groups that have **"manage"** permission on the object have full access to it including the ability to write to the object and to change its permissions.
Publicly listable	Anyone can have **read-only** access to the public metadata of the object (stored at `/public`) and its playable offerings. Only members of Access Groups that have **"access"** permission on the object can read the non-public metadata and offerings, and only those with **"manage"** permission can write to the object or change its permissions.
Public	Anyone has **read-only** access to the object (both public and private parts). Only those with **"manage"** permission can write to the object or change its permissions.

Object Group Permissions

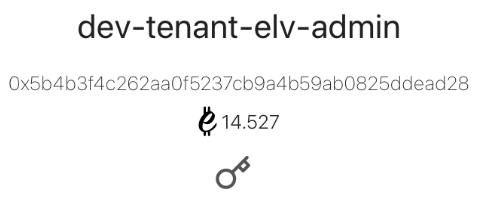
AWAITING INFO ON RECENT CONTRACT CHANGES.

User Account / Profile

Content Fabric Accounts gain access via a **Private Key**. Do not share your Private Key. Keep a copy of it in a safe place. **If you lose your Private Key, it cannot be recovered.**

In addition to a Private Key, Accounts also have a **Public Key** and an **Address**.

You can view these items in the Fabric Browser by clicking on the **ELUV.IO** logo then clicking **Profile**. When you first go to this page, only your Address will be visible:



dev-tenant-elv-admin

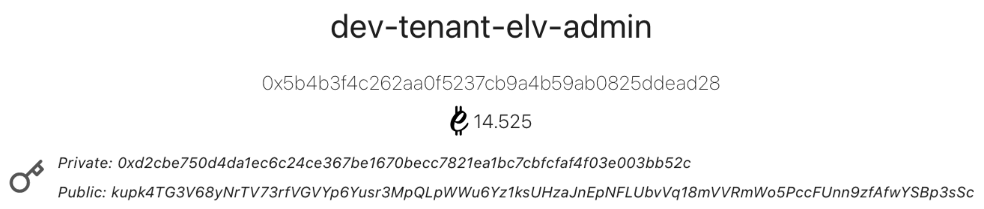
0x5b4b3f4c262aa0f5237cb9a4b59ab0825ddead28

14.527

The Address is used for funding your Account. If you ever need an administrator to add credits to your Account, you will need to let them know your Address.

A Content Fabric Address consists of `0x` followed by 40 hex digits (`0-9` and/or `a-f`).

To view your **Public** and **Private Keys**, click on the key icon.



dev-tenant-elv-admin

0x5b4b3f4c262aa0f5237cb9a4b59ab0825ddead28

14.525

Private: 0xd2cbe750d4da1ec6c24ce367be1670becc7821ea1bc7cbfcfaf4f03e003bb52c

Public: kupk4TG3V68yNrTV73rfVGVYp6Yusr3MpQLpWWu6Yz1ksUHzaJnEpNFLUbvVq18mVVRmWo5PccFUnn9zfAfwYSBp3sSc

A Private Key consists of `0x` followed by 64 hex digits.

A Public Key is a long alphanumeric string of 90 or more characters.

The Fabric Browser Profile page can also be used to edit the name and image displayed for your Account.

Metadata

(WIP: public / private metadata)

Playout (Streaming) Formats

The two most common formats for streaming media over the internet are **HLS** and **DASH**. The Eluvio Content Fabric supports both, and you can offer both formats using a single Offering.

	HLS	**DASH**
Full name	HTTP Live Streaming *(note that format is not restricted to live streams, VoD is supported as well)*	Dynamic Adaptive Streaming over HTTP
Developed by	Apple	Moving Picture Experts Group (MPEG)
Playlist/manifest file extension	.m3u8	.mpd
Playlist/manifest MIME type	application/vnd.apple.mpegurl	application/dash+xml
Standards doc(s)	https://datatracker.ietf.org/doc/html/rfc8216	https://www.iso.org/obp/ui#iso:std:iso-iec:23009:-1:ed-5:v1:en
Pros	● More widely supported	● Open source ● Codec agnostic ● Manifests can be more compact that HLS playlists
Cons	● Only supports H.264 and H.265 video codecs ● Playlist files can be large for longer duration media	● Not directly supported by iOS, AppleTV, Safari
Content Fabric-supported DRM Formats	● Fairplay ● Sample-AES ● AES-128	● Widevine

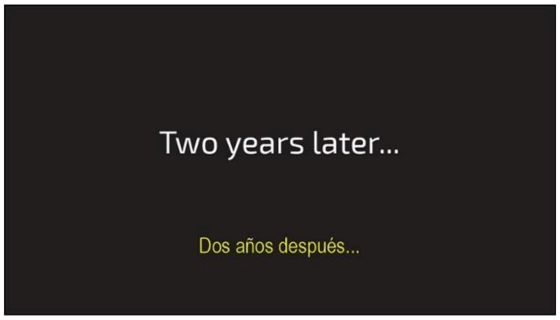
Audio streams and channels

Audio mixing / channel selection

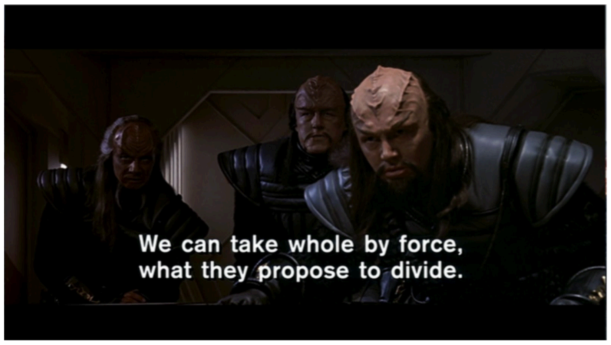
Subtitles

Subtitles - Forced

Forced Subtitles are ones that translate audio dialogue or text seen on screen (e.g. a street sign or a newspaper headline) that may not be understood by the viewer - e.g. for a Spanish-speaking viewer, the following is an example translating on-screen English text:



They are most commonly used for scenes where the audio dialogue is in a language not expected to be understood by the viewer (e.g. fictional languages like Elvish, Klingon or Dothraki, or any language other than that used in the rest of the sound track). The example below is for English-speaking viewers, translating lines spoken in Klingon:



Forced subtitles are usually quite sparse, as they do not translate all spoken lines in a program. A given piece of content may only have a few lines.

There may be multiple sets of forced subtitles (often one for every audio language track) and they are meant to be used together with the matching audio track - e.g. if someone has chosen the English audio track, then an English forced subtitle track will be used (if one exists) and likewise if someone has selected a Spanish audio track, then Spanish forced subtitles will be used. In Offerings that have multiple audio and/or subtitle streams, every stream should be assigned a language code to help players automatically select the appropriate streams.

They are called 'forced' because they are supposed to be shown even if the viewer has not turned on subtitles. For this reason, they are also not supposed to show as a choice in UI subtitle pickers.

Instead, when a user switches audio track to a particular language, if subtitles are turned OFF, then forced subtitles for the matching language should automatically be used.

When a user selects a subtitle track, then no forced subtitle track should be used at all. However, 'normal' subtitle tracks often include the forced subtitles anyway. This is recommended by Netflix as best practice:

> **On our service, Forced Narrative subtitles are only displayed if Subtitles and CC are set to "off" in the user's playback settings. When the user activates a full Subtitle or SDH/CC file, the FN subtitle does not display and for this reason, we require that all Forced Narrative events are also included in each full Subtitle and SDH/CC file.**

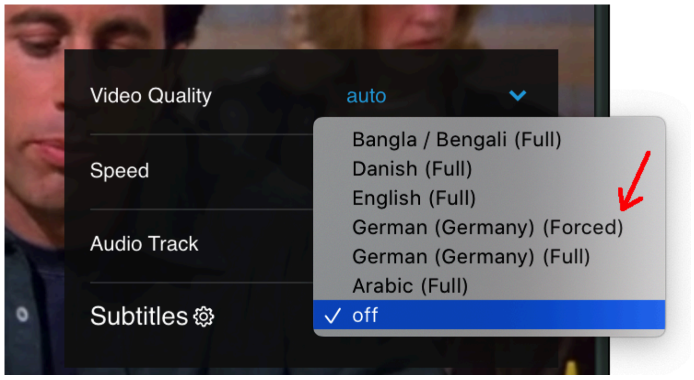
(from Understanding Forced Narrative Subtitles – Netflix | Partner Help Center)

Note that while the terms 'subtitles' and 'captions' are often used interchangeably, strictly speaking there is a difference in definition:

- *subtitles* are meant for people who don't understand the audio language (but can still hear)
- *captions* are meant for people who cannot hear, and may include additional text descriptions of audio events, e.g.:

In practice however this distinction is not always observed. Currently in the fabric we do not distinguish between subtitles and captions, and internally the two terms are used interchangeably.

The Eluvio Content Fabric supports Forced Subtitle tracks. However, player support for them is uneven, particularly among open source projects. Depending on what player is used, Forced Subtitles may be treated (incorrectly) as just another normal subtitle stream and shown alongside them:



Library Metadata

***TODO**: notes on automation support, storing ABR Profile, additional offering specs, types and groups for SimpleIngest.js*

Streamlining Commands with Configuration Files

It is possible to use a **Configuration File** to automatically set environment variables and (some) command line options for you. These are JSON files with information specific to your Tenancy.

Safeguarding your information

As Configuration Files often contain your Private Key and other potentially sensitive information you should limit who can read them, e.g. with:

```
chmod 600 $YOUR_CONFIGURATION_FILE_NAME
```

Usage

There are two ways to use a Configuration File:

- Include `--confs $PATH_TO_YOUR_CONFIG_FILE` in your command line
- Set the `ELVUTILS_CONFIG` environment variable

```
# using --confs
node ListLibraries.js --confs $PATH_TO_YOUR_CONFIGURATION_FILE

# using environment variable
export ELVUTILS_CONFIG=$PATH_TO_YOUR_CONFIGURATION_FILE
node ListLibraries.js
```

When you run most utility scripts, if a Configuration File is specified using either of the above methods you will see a notification at the beginning of the script's output, e.g.:

```
ELVUTILS_CONFIG env var found, loading config from: /myFolder/myConfig.json
```

There are a few utilities that do not take any options at all. These are not affected by Configuration Files.

Format (basic)

A basic Configuration File has one entry named `"defaults"` with a number of sub-entries:

`/elv-utils-js/example-files/config_basic.json`

```
{
  "defaults": {
    "abrProfile":        "$_ElvUtilsDir_/example-files/abr_profile_both.json",
    "FABRIC_CONFIG_URL":"https://... (Your Fabric configiation URL)",
    "groupAddress":      "0x... (your Content Admins group address)",
    "masterLib":         "ilib... (your Master library ID)",
    "masterType":        "iq__... (your Master content type ID)",
    "mezLib":            "ilib... (your Mezzanine library ID)",
    "mezType":           "iq__... (your Mezzanine content type ID)",
    "PRIVATE_KEY":       "0x... (your private key)"
  }
}
```

- Sub-entries with names that are in ALL_CAPS_WITH_UNDERSCORES (aka "SCREAMING_SNAKE_CASE") will be used to set environment variables. In the example above, `FABRIC_CONFIG_URL` and `PRIVATE_KEY` will be set while the script is running. More specifically, to be used as an environment variable, the entry name must:
  - Contain only capital letters and underscores
  - Contain at least one underscore
  - Begin and end with a capital letter
  - Not be "ELVUTILS_CONF"
- Sub-entries with names that do not not qualify as environment variables will be used to fill in any missing command line options with the same name (or an option alias that matches the name)
- The values stored for sub-entries must be either text or arrays of text. This is the case even for numeric options such as `--ethContractTimeout` (e.g. `"ethContractTimeout": "30"`)
- Arrays of text are used for command line options that take multiple values, e.g. `--files`
- Dollar signs (`$`) have a special meaning in values - they are used to specify substitution variables.
  - For example, "`$_ElvUtilsDir_`" in the value at `/defaults/abrProfile` will be substituted with the path to your `/elv-utils-js` directory.
  - If you actually need to include a dollar sign in a value, use two in a row (`$$`). These will not be processed as substitution variables, and will be replaced with single dollar signs before use.

Command line option handling

To see a command's list of options and their aliases, run it with the `--help` option, e.g.:

```
node MasterCreate.js --help
```

Sample (partial) output:

```
--libraryId, --masterLib, --master-lib,           Library ID for new master (should start with
--library-id                                      'ilib')                     [string] [required]
--metadata                                        JSON string for metadata or path to JSON file
                                                  containing the metadata of master object
                                                                                        [string]
--storeClear, --store-clear                       Store uploaded/copied files unencrypted
                                                                                       [boolean]
--streams                                         JSON string or path to JSON file containing
                                                  stream specifications for variant in new
                                                  master                                [string]
--type, --masterType, --master-type               Name, object ID, or version hash of content
                                                  type for new master         [string] [required]
```

The above shows that `--type` has 2 aliases: `--masterType` and `--master-type`. All three are synonymous.

If you use the `config_basic.json` file and run `MasterCreate.js` *without* `--type` *(or `--masterType` or `--master-type`)*, then the value stored in the Configuration File under `/defaults/masterType` will be used. Similarly, if `--libraryId` is missing, the value under `/defaults/masterLib` will be used, allowing you to shorten the `MasterCreate.js` command (and skip having to set environment variables):

```
node MasterCreate.js \
  --title "$YOUR_TITLE" \
  --files $PATH_TO_YOUR_FILE
```

Advanced usage: multiple configuration files

You can create multiple Configuration Files for different scenarios and switch between them - for example, if you ingest from S3, you can create a second file with additional entries:

```
{
  "defaults": {
    "abrProfile":       "$_ElvUtilsDir_/example-files/abr_profile_both.json",
    "FABRIC_CONFIG_URL":"https://... (Your Fabric configuration URL)",
    "groupAddress":     "0x... (your Content Admins group address)",
    "masterLib":        "ilib... (your Master library ID)",
    "masterType":       "iq__... (your Master content type ID)",
    "mezLib":           "ilib... (your Mezzanine library ID)",
    "mezType":          "iq__... (your Mezzanine content type ID)",
    "PRIVATE_KEY":      "0x... (your private key)",
    "AWS_BUCKET":       "your-bucket-name",
    "AWS_KEY":          "your-aws-key",
    "AWS_REGION":       "your-aws-region",
    "AWS_SECRET":       "your-aws-secret",
    "s3Reference":      "true"
  }
}
```

If you do set up multiple Configuration Files, it is recommended to use the **`--confs`** option instead of the **`ELVUTILS_CONFIG`** environment variable.

If you do use the **`ELVUTILS_CONFIG`** environment variable and work with multiple Configuration Files, then to help prevent mistakes you may wish to change your command line system prompt to show the value of **`$ELVUTILS_CONFIG`**. For example, on recent versions of OS X (which use the ZSH shell) you can insert the following in your **`~/.zshrc`** file:

```
setopt prompt_subst

function precmd() {
  if [ -z "$ELVUTILS_CONFIG" ];
  then
    PROMPT="%n@%m %1~  %# "
  else
    CONFIG_FILENAME=$(basename $ELVUTILS_CONFIG)
    PROMPT="%n@%m %1~ %B%F{yellow}$CONFIG_FILENAME%f%b! "
  fi
}
```

It is also possible to use multiple Configuration Files in a single command, either by using both **`--confs`** and **`ELVUTILS_CONFIG`** or by listing multiple files after **`--confs`**. The files will be merged together, with later files taking precedence over earlier files if the same entry exists in multiple files. (If both **`--confs`** and **`ELVUTILS_CONFIG`** are used then the file specified by **`ELVUTILS_CONFIG`** is loaded first, then the file(s) listed after **`--confs`** are merged on top of it)

Format (advanced)

In addition to the `"defaults"` section, a Configuration File can have a `"presets"` section. This section holds groups of settings that can be imported into the `"defaults"` section by adding a `"presets_add"` entry.

The example file `/elv-utils-js/example-files/config_complex.json` uses Presets. Note that a Preset can also import other Presets by specifying a `"presets_add"` entry - for example, the ***"ingest_s3"*** Preset combines the ***"ingest"*** and ***"s3"*** presets:

`/elv-utils-js/example-files/config_complex.json`

```
{
  "defaults": {
    "presets_add": ["client", "tenant",  "ingest"]
  },

  "presets": {

    "client": {
      "ethContractTimeout": "30"
    },

    "tenant": {
      "FABRIC_CONFIG_URL": "https://demov3.net955210.contentfabric.io/config",
      "PRIVATE_KEY":       "your-private-key",
      "groupAddress":      "0x.. your-content-admins-group-contract-address",
      "masterLib":         "ilib... your-master-library-id",
      "masterType":        "iq__... your-master-content-type-id",
      "mezLib":            "ilib... your-mezzanine-library-id",
      "mezType":           "iq__... your-channel-content-type-id"
    },

    "s3": {
      "AWS_BUCKET": "your-bucket-name",
      "AWS_KEY":    "your-aws-key",
      "AWS_REGION": "your-aws-region",
      "AWS_SECRET": "your-aws-secret"
    },

    "abr_profiles": {
      "abr_profile_dir":                "$_ElvUtilsDir_/example-files",
      "abrp_both":                      "$abr_profile_dir/abr_profile_both.json",
      "abrp_no_drm_store_encrypted":   "$abr_profile_dir/abr_profile_no_drm_store_encrypted.json",
      "abrp_no_drm_store_unencrypted":"$abr_profile_dir/abr_profile_no_drm_store_unencrypted.json",
      "abrp_drm_all":                  "$abr_profile_dir/abr_profile_drm.json",
      "abrp_drm_strict":               "$abr_profile_dir/abr_profile_drm_strict.json"
    },

    "ingest": {
      "presets_add": ["abr_profiles"],
      "abrProfile": "$abrp_both"
    },

    "ingest_s3": {
      "presets_add":      ["ingest","s3"]
    },

    "ingest_s3_reference": {
      "presets_add": ["ingest_s3"],
      "s3Reference": "true"
    },

    "use_example_video_file": {
      "files":            "$_ElvUtilsDir_/example-files/video.mp4"
    },

    "airtable" : {
      "AIRTABLE_API_KEY": "your-airtable-access-token",
      "baseId":           "your-airtable-base-ID"
    },

    "debug": {
      "ELVUTILS_THROW": "1",
      "debug":          "true"
    },

    "lib_master": {
      "libraryId": "$masterLib"
    },

    "lib_mez": {
      "libraryId": "$mezLib"
    },

    "type_master": {
      "type": "$masterType"
    },

    "type_mez": {
      "type": "$mezType"
    }
  }
}
```

Note also that dollar signs can be used for more than just the pre-defined `$_ElvUtilsDir_` substitution variable. They are also used to create cross-references to other entries. In the example above, the ***"abr_profiles"*** Preset has an entry of `"abrp_both" : "$abr_profile_dir/abr_profile_both.json"`. The `$abr_profile_dir` part of that entry's value will be swapped out and replaced by the value of the ***"abr_profile_dir"*** entry before being used by the script. (The ***"abr_profile_dir"*** entry's value itself contains `$_ElvUtilsDir_`, which will be replaced by the actual path to `/elv-utils-js`.)

Advanced usage: --presets

When you run a command, you can also cause one or more Presets to be included by using `--presets` followed by the Preset name(s). In the `config_complex.json` file, there are a number of Presets that are not imported by the ***"defaults"*** section, for example:

```
    "lib_master": {
      "libraryId": "$masterLib"
    },

    "lib_mez": {
      "libraryId": "$mezLib"
    },
```

If you were using a similar Configuration File, you could list the Objects in your Master Library with:

```
node LibraryListObjects.js --presets lib_master
```

Since `--libraryId` was not supplied on the command line, the value under `/presets/lib_mez/libraryId/` will be used (`$mezLib` winds up cross-referencing to `/presets/tenant/mezLib`). See [Reference: Configuration File syntax and processing]() for more details on how presets and cross-references are handled.

Using a utility script as a module

AirTable integration

Updating your copy of elv-utils-js

If you need to update to the latest version of `elv-utils-js`, you can do so with:

```
cd elv-utils-js
git pull
npm install
npm audit fix
```

Workflow examples

All the command line examples below use a dollar sign followed by an ITEM_NAME in all caps to indicate information that you need to substitute in (e.g. ***$MASTER_TYPE***, ***$FILENAME***).

Four of these (***$MASTER_TYPE***, ***$MEZ_TYPE***, ***$MASTER_LIB***, ***$MEZ_LIB***) are so frequently used that for convenience, you may wish to set these up as environment variables with the values you looked up in the previous section, e.g.: *(replace with your own IDs)*

```
export MASTER_TYPE=iq__2tfLjovW8zMN9Yh6eLmwynX1Cbip
export MEZ_TYPE=iq__8SLzhEyJWiJ41BPezhswG56MUwL
export MASTER_LIB=ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
export AWS_SECRET=ilib29dvmbN91uyXRwcMX88CAs8q2zeT
```

Test ingest - small local file (no DRM)

The following is suitable for ingesting short videos (several minutes or less) with a single audio stream, where both audio and video are included in a single file. The example ABR Profile used sets a top resolution of 1080p and specifies no DRM.

Because **`--wait`** is specified, the **`MezCreate.js`** command will wait for transcoding to finish and automatically finalize changes. It will however tie up your command line while transcoding is taking place.

1. Create the Master:

```
node MasterCreate.js \
  --libraryId $MASTER_LIB \
  --type $MASTER_TYPE \
  --title "$YOUR_TITLE" \
  --files $PATH_TO_YOUR_FILE
```

2. Copy the version hash output by the **`MasterCreate.js`** script. This will need to be substituted in for ***$MASTER_VERSION_HASH*** in the next command.
3. Create the Mezzanine (and wait for transcoding to finish)

```
node MezCreate.js \
  --libraryId $MEZ_LIB \
  --type $MEZ_TYPE \
  --title "$YOUR_TITLE" \
  --masterHash $MASTER_VERSION_HASH \
  --abrProfile ../example-files/abr_profile_no_drm_store_encrypted.json \
  --wait
```

Typical Ingest - longer file / S3 / DRM / shared with Admin group

The following is a typical ingest workflow with the following parameters:

- File(s) are longer duration
- Offering will provide HD resolution (1080p)
- Master file(s) are hosted on AWS S3, and not copied to Content Fabric
- Playout uses 'strict' DRM *(Fairplay or Widevine only)*
- Master and Mezzanine Objects are managed by your 'Content Admins' group

For longer duration files, tying up the command line until an ingest finishes transcoding is impractical. Usually you will run the **`MezCreate.js`** command without the **`--wait`** option. You can check the progress and status of ingests with the **`MezJobStatus.js`** command, then run the command again with the **`--finalize`** option once finished.

1. Create the Master:

```
node MasterCreate.js \
  --libraryId $MASTER_LIB \
  --type $MASTER_TYPE \
  --title "$YOUR_TITLE" \
  --s3Reference \
  --files $S3_PATH_TO_YOUR_FILE
```

2. Copy the Object ID output by the **`MasterCreate.js`** script. This will need to be substituted in for ***`$MASTER_OBJECT_ID`*** in the next command

3. Grant **`manage`** permission for the Master to your **`Content Admins`** Access Group:

```
node ObjectAddGroupPerm.js \
  --objectId $MASTER_OBJECT_ID \
  --groupAddress $ACCESS_GROUP_ADDRESS \
  --permission manage
```

4. Copy the Version Hash output by the **`MasterCreate.js`** script from step 1. This will need to be substituted in for ***`$MASTER_VERSION_HASH`*** in the next command.

5. Create the Mezzanine:

```
node MezCreate.js \
  --libraryId $MEZ_LIB \
  --type $MEZ_TYPE \
  --title "$YOUR_TITLE" \
  --masterHash $MASTER_VERSION_HASH \
  --abrProfile ../example-files/abr_profile_drm_strict.json
```

6. Copy the Object ID output by the `MasterCreate.js` script. This will need to be substituted in for `$MEZ_OBJECT_ID` in the rest of the commands.

7. Check on the transcoding progress:

```
node MezJobStatus.js \
  --objectId $MEZ_OBJECT_ID
```

8. Once the job is finished, finalize the Mezzanine:

```
node MezJobStatus.js \
  --objectId $MEZ_OBJECT_ID \
  --finalize
```

9. Grant `manage` permission for the Mezzanine to your `Content Admins` Access Group:

```
node ObjectAddGroupPerm.js \
  --objectId $MEZ_OBJECT_ID \
  --groupAddress $ACCESS_GROUP_ADDRESS \
  --permission manage
```

10. Set the Object Permission on the Mezzanine to `viewable`:

```
node ObjectSetPermission.js \
  --objectId $MEZ_OBJECT_ID \
  --permission viewable
```

Use a single object as both Master and Mezzanine

Create both a normal and a watermarked Offering during ingest

Undo a change (revert to an earlier version)

Create a master from S3 (link files)

When working with masters stored in AWS S3, usually you will want to link to the master file(s) instead of copying them to the Content Fabric. This avoids the time needed to copy, reduces storage used by your tenancy, and keeps access to your master assets as limited as possible. Once your transcodes are finalized you can move the source files to S3 Glacier, remove them from S3 altogether, or keep them in your S3 bucket but revoke access for the S3 credentials used during ingest.

Keep in mind that if you need to re-transcode from the master files (for example, to increase the top resolution/bit rate) you will need to restore them to the same path in the same S3 bucket first.

Before running the command, you must set the environment variables to specify your AWS region, bucket, and credentials:

```
export AWS_REGION=     (your AWS region)
export AWS_BUCKET=     (your AWS bucket name)
export AWS_KEY=AK...   (your AWS S3 key)
export AWS_SECRET=...  (your AWS S3 secret)
```

NOTE: these variables must also be set whenever you run commands that need to access the contents of linked files: `FilesProbe.js`, `MasterInit.js`, `MasterUpdateSources.js`, *and* `MezCreate.js`. *The Content Fabric does not store your AWS credentials.*

The paths to use for the `--files` option should take the form:

`s3://BUCKET_NAME/FILE_PATH_WITHIN_BUCKET`

For example, if my bucket name were "ingest" and my file "master.mp4" was in the top level of the bucket, then it would be `s3://ingest/master.mp4` (Note that my `AWS_BUCKET` environment variable would also be set to `ingest`)

The command to create the Master is then:

```
node MasterCreate.js \
  --libraryId $MASTER_LIB \
  --type $MASTER_TYPE \
  --title "$YOUR_TITLE" \
  --s3Reference \
  --files $FILE_1_S3_PATH $FILE_2_S3_PATH...
```

Create a master from S3 (copy files)

If you wish to copy the file from S3 into the Content Fabric, use the `--s3Copy` option. This may take a while, depending on file size and proximity of the AWS S3 server to the Content Fabric node.

If you are in a different geographic area than the S3 region that hosts your bucket, you can use the --elvGeo option to choose a Content Fabric node that is closer to the S3 servers.

s3://BUCKET_NAME/bbb_sunflower_1080p_60fps_stereo_abl.mp4

```
node MasterCreate.js \
  --libraryId $MASTER_LIB \
  --type $MASTER_TYPE \
  --title "$YOUR_TITLE" \
  --s3Copy \
  --files $S3_PATH_TO_YOUR_FILE
```

Duplicate an offering

Add DRM to an offering

Change offering DRM to 'strict'

Remove DRM from an offering

Replace a stream (revised master file provided after initial ingest)

Update stream `'video'` in Variant named `'default'` to use a new file stored on S3, then redo the stream in the Offering (also named `'default'`)

1. Add file to Master

```
node utilities/FilesAdd.js \
  --objectId $MASTER_OBJECT_ID \
  --files $FILE_S3_URL \
  --s3Reference
```

2. Probe file and update info in Master

```
node utilities/MasterUpdateSources.js \
  --objectId $MASTER_OBJECT_ID \
  --files $FILENAME
```

3. Change stream in the Variant to use the new file as source (NOTE: the version hash output by this command will need to be substituted in for `$NEW_MASTER_HASH` in the next step).

```
node utilities/VariantEditStream.js \
  --objectId $MASTER_OBJECT_ID \
  --streamKey video \
  --file $FILENAME
```

4. Ingest to existing Offering `'default'` in existing Mezzanine object, transcoding only 1 stream and preserving other previously ingested streams

```
node utilities/MezCreate.js \
  --masterHash $NEW_MASTER_HASH \
  --objectId $MEZ_OBJECT_ID \
  --abrProfile utilities/example-files/abr_profile_drm.json \
  --streamKeys video \
  --keepOtherStreams
```

5. Check status of transcode

```
node utilities/MezJobStatus.js \
  --objectId $MEZ_OBJECT_ID
```

6. Finalize as usual once transcoding is finished.

```
node utilities/MezJobStatus.js \
  --objectId $MEZ_OBJECT_ID \
  --finalize
```

Additional audio language file provided after ingest

Add a new stream `'audio-fr'` to Variant `'default'`, using a new file on S3, then update the Offering (also named `'default'`)

1. Add file to Master

```
node utilities/FilesAdd.js \
  --objectId $MASTER_OBJECT_ID \
  --files $FILE_S3_URL \
  --s3Reference
```

2. Probe file and update info in Master

```
node utilities/MasterUpdateSources.js \
  --objectId $MASTER_OBJECT_ID \
  --files $FILENAME
```

3. Add new stream to Variant, using the new file as source (NOTE: the version hash output by this command will need to be substituted in for `$NEW_MASTER_HASH` in the next step)

```
node utilities/VariantAddStream.js \
  --objectId $MASTER_OBJECT_ID \
  --streamKey audio-fr \
  --file $FILENAME \
  --streamIndex 0 \
  --language fr \
  --label Français
```

4. Ingest to existing Mezzanine / Offering, transcoding only 1 stream (preserving the other streams)

```
node utilities/MezCreate.js \
  --masterHash $NEW_MASTER_HASH \
  --objectId $MEZ_OBJECT_ID \
  --abrProfile example-files/abr_profile_both.json \
  --streamKeys audio-fr \
  --keepOtherStreams
```

5. Check transcode status

```
node utilities/MezJobStatus.js \
  --objectId $MEZ_OBJECT_ID
```

6. Finalize as usual once transcoding is finished.

```
node utilities/MezJobStatus.js \
  --objectId $MEZ_OBJECT_ID \
  --finalize
```

Change a single metadata field

Reference: elv-utils-js /utilities

All the command line examples use a dollar sign followed by an ITEM_NAME in all caps to indicate information that you need to substitute in (e.g. `$MASTER_TYPE`, `$FILENAME`).

They also assume that you have set required environment variables and are currently in the `elv-utils-js/utilities` directory.

All the scripts in `elv-utils-js/utilities/` except `TextConvertForCmd.js` have a `--help` option that will show all the available options for that script.

FilesAdd.js

- Adds files to an object
- The files to add may be local or on S3
- If local, the files will be uploaded to the fabric.
- If on S3, the files may either be copied to the fabric (`--s3Copy`) or only linked (`--s3Reference`)
- Files are stored encrypted by default, use `--storeClear` if you wish files to use unencrypted storage. (does not apply if `--s3Reference` is used, in which case the files are not stored in the fabric at all)

```
node FilesAdd.js \
  --objectId $MASTER_OBJECT_ID \
  --files audio-french.mp4
```

Sample output:

```
Add file(s) to object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Getting write token...
New write token:
        tqw__HSRvTnR3BpmgffhUQgNyc9KeJUA8AGLLbUqAPDcqNunKTbhf4oVrmqYnxmaV441b3pu7C3V6F8JWLCqXtLU
       node URL: http://localhost:8008
audio-french.mp4: 0.0%
audio-french.mp4: 91.7%
audio-french.mp4: 100.0%
Finalizing object...
Finalized, new version hash:
        hq__7EguUjRw2TCPt4jcWAtrRimxxN4ce4Cq9wvFAhnPTUyhjnrC2vV68s3sWkaHiTdLokDsx4DfpC
Waiting for publishing to finish and new object version to become available...
New object version now available

File(s) added.
New version hash: hq__7EguUjRw2TCPt4jcWAtrRimxxN4ce4Cq9wvFAhnPTUyhjnrC2vV68s3sWkaHiTdLokDsx4DfpC


Done.
```

FilesDownload.js

- Download file(s) from an object.
- Will exit with an error if file(s) already exist in the target directory or if file is not found in the object.
- Note that linked S3 files (i.e. those added to an object using `--s3Reference`) cannot be downloaded.
- If you wish to download files from an old Version of an Object, use `--versionHash` instead of `--objectId`.

```
node FilesDownload.js \
  --objectId  $MASTER_OBJECT_ID \
  --filePaths     audio-english.mp4
```

Sample output:

```
Download file(s) from object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Decoding object id from hash
        hq__3HZ5HsVQPFgQPEVtmkswvHEBBnU8fw9nhyBKMtJhyatirWmvtSsbZBzpxFSbCKPjrq6svWaCnc...
Found: iq__ho5NsUpev3AQv4XY1WvN8AwBXDz
Reading metadata from version
        hq__3HZ5HsVQPFgQPEVtmkswvHEBBnU8fw9nhyBKMtJhyatirWmvtSsbZBzpxFSbCKPjrq6svWaCnc...
Retrieving metadata path '/files' from version
        hq__3HZ5HsVQPFgQPEVtmkswvHEBBnU8fw9nhyBKMtJhyatirWmvtSsbZBzpxFSbCKPjrq6svWaCnc...
audio-english.mp4: 9%
audio-english.mp4: 17%
audio-english.mp4: 26%
audio-english.mp4: 35%
audio-english.mp4: 43%
audio-english.mp4: 52%
audio-english.mp4: 60%
audio-english.mp4: 69%
audio-english.mp4: 78%
audio-english.mp4: 86%
audio-english.mp4: 95%
audio-english.mp4: 100%

Done.
```

FilesProbe.js

- Probes files for media and returns information on any streams found
- If you wish to probe files in an old Version of an Object, use `--versionHash` instead of `--objectId`.

```
node FilesProbe.js \
  --objectId  $MASTER_OBJECT_ID \
  --files     audio-english.mp4
```

Sample output:

```
Get media info for file(s) in iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
{
  "audio-english.mp4": {
    "container_format": {
      "duration": 718.9970092773438,
      "filename": "audio-english.mp4",
      "format_name": "mov,mp4,m4a,3gp,3g2,mj2",
      "start_time": 0
    },
    "streams": [
      {
        "type": "StreamAudio",
        "channel_layout": "stereo",
        "channels": 2,
        "sample_rate": 48000,
        "codec_name": "aac",
        "language": "",
        "side_data_list": [],
        "tags": {},
        "bit_rate": 127317,
        "duration": 718.997,
        "duration_ts": 34511856,
        "frame_count": 33704,
        "max_bit_rate": 0,
        "start_pts": 0,
        "start_time": 0,
        "time_base": "1/48000"
      }
    ]
  }
}

Done.
```

ImageSet.js

- Sets the image displayed for an Object in the Fabric Browser to an image file within the Object
- The image must have already been added to the Object (use the **`FilesAdd.js`** script if needed)
- If you wish to clear the displayed image, use **`--clear`** instead of **`--filePath`**.

```
node ImageSet.js \
  --objectId  $EXISTING_OBJECT_ID \
  --filePath  $FILE_PATH_WITHIN_OBJECT
```

Sample output:

```
Set display image for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz to /images/Poster.png

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Checking that file exists...
Getting write token...
New write token:
        tqw__HSRvTnR3BpmgffhUQgNyc9KeJUA8AGLLbUqAPDcqNunKTbhf4oVrmqYnxmaV45Hm8qjz9zWFs3eMxDQdAvU
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__LyeakvgRN5Fo8PgTZGuxdRArmPsyv7z51VKpGXXDBft4HyVsr982bggrnEfPikFuFkwzQRuJ6t
Waiting for publishing to finish and new object version to become available...
New object version now available
Version Hash: hq__LyeakvgRN5Fo8PgTZGuxdRArmPsyv7z51VKpGXXDBft4HyVsr982bggrnEfPikFuFkwzQRuJ6t

Done.
```

LibraryInfo.js

- Lists information for a single library.

```
node LibraryInfo.js \
  --libraryId $MEZ_LIB
```

Sample output:

```
Get info for library ilib29dvmbN91uyXRwcMX88CAs8q2zeT

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
{
  "contractMetadata": {},
  "latestHash": "hq__BaC7ZA6fiuX6Sr6426QezW5hLDSJDfFx9DVZURVYUUQCupp5dLHSLe4iHCjm3iqfkGZ12BvaNT",
  "metadata": {
    "commit": {
      "author": "dev-tenant-elv-admin",
      "author_address": "0x5b4b3f4c262aa0f5237cb9a4b59ab0825ddead28",
      "message": "Create library",
      "timestamp": "2023-01-19T23:36:38.209Z"
    },
    "elv": {
      "media": {
        "drm": {
          "fps": {
            "cert":
       "MIIExzCCA6+gAwIBAgIIHyfkXhxLHC4wDQYJKoZIhvcNAQEFBQAwfzELMAkGA1UEBhMCVVMxEzARBgNVBAoMCkFwc
       GxlIEluYy4xJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MTMwMQYDVQQDDCpBcHBsZSBLZXk
       gU2VydmljZXMgQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkwHhcNMjAwOTEyMDMzMjI0WhcNMjIwOTEzMDMzMjI0WjBgM
       QswCQYDVQQGEwJVUzETMBEGA1UECgwKRWx1dmlvIEluYzETMBEGA1UECwwKMktIOEtDM01NWDEnMCUGA1UEAwweRmF
       pclBsYXkgU3RyZWFtaW5nOiBFbHV2aW8gSW5jMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDslbBURB6gj07g7
       VrS7Ojixe7FNZOupomcZt+mtMvyavjg7X7/T4RccmKUQxOoMLKCJcQ6WrdHhIpN8+bciq7lr0mNzaN467zREiUNYOp
       kVPi13sJLieY2m2MEPOQTbIl52Cu1YyH+4/g1dKPmeguSnzZRo36jsCGHlJBjHq0jkQIDAQABo4IB6DCCAeQwDAYDV
       R0TAQH/BAIwADAfBgNVHSMEGDAWgBRj5EdUy4VxWUYsg6zMRDFkZwMsvjCB4gYDVR0gBIHaMIHXMIHUBgkqhkiG92N
       kBQEwgcYwgcMGCCsGAQUFBwICMIG2DIGzUmVsaWFuY2Ugb24gdGhpcyBjZXJ0aWZpY2F0ZSBieSBhbnkgcGFydHkgY
       XNzdW1lcyBhY2NlcHRhbmNlIG9mIHRoZSB0aGVuIGFwcGxpY2FibGUgc3RhbmRhcmQgdGVybXMgYW5kIGNvbmRpdGl
       vbnMgb2YgdXNlLCBjZXJ0aWZpY2F0ZSBwb2xpY3kgYW5kIGNlcnRpZmljYXRpb24gcHJhY3RpY2Ugc3RhdGVtZW50c
       y4wNQYDVR0fBC4wLDAqoCigJoYkaHR0cDovL2NybC5hcHBsZS5jb20va2V5c2VydmljZXMuY3JsMB0GA1UdDgQWBBR
       4jerseBHEUDC7mU+NQuIzZqHRFDAOBgNVHQ8BAf8EBAMCBSAwOAYLKoZIhvdjZAYNAQMBAf8EJgFuNnNkbHQ2OXFuc
       3l6eXp5bWFzdmdudGthbWd2bGE1Y212YzdpMC4GCyqGSIb3Y2QGDQEEAQH/BBwBd252bHhlbGV1Y3Vpb2JyZW4yeHZ
       lZmV6N2Y5MA0GCSqGSIb3DQEBBQUAA4IBAQBM17YYquw0soDPAadr1aIM6iC6BQ/kOGYu3y/6AlrwYgAQNFy8DjsQU
       oqlQWFuA0sigp57bTUymkXEBf9yhUmXXiPafGjbxzsPF5SPFLIciolWbxRCB153L1a/Vh2wg3rhf4IvAZuJpnml6SS
       g5SjD19bN+gD7zrtp3yWKBKuarLSjDvVIB1SoxEToBs3glAEqoBiA2eZjikBA0aBlbvjUF2gqOmZjZJ7dmG1Tos2Zd
       4SdGL6ltSpKUeSGSxyv41aqF83vNpymNJmey2t2kPPtC7mt0LM32Ift3AkAl8Za9JbV/pOnc95oAfPhVTOGOI+u2Bu
       B2qaKWjqHwkfqCz4A"
          }
        }
      }
    },
    "name": "dev-tenant - Title Mezzanines",
    "public": {
      "name": "dev-tenant - Title Mezzanines"
    }
  },
  "objectId": "iq__29dvmbN91uyXRwcMX88CAs8q2zeT",
  "type": ""
}

Done.
```

LibraryListObjects.js

- List objects visible to the current user (i.e. the current private key) for a single library
- By default, shows only object IDs, but options are available to show additional fields (use `--help` for a full list)

```
node LibraryListObjects.js \
  --libraryId $MEZ_LIB \
  --name \
  --date
```

Sample output:

```
List objects in library ilib29dvmbN91uyXRwcMX88CAs8q2zeT

Getting list of objects...
Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Found 2 object(s)
Retrieving metadata path '/commit' from object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Retrieving metadata path '/commit' from object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...

objectId                         name              commitDate
iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA Big Buck Bunny MEZ 2023-01-25T00:54:40.278Z
iq__ksXLW12ZAAg7PcLS4gKLUPPRUin  Meridian MEZ       2023-01-20T00:06:50.263Z


Done.
```

ListAccessGroups.js

- Lists names and addresses for Access Groups visible to the current user (i.e. the current private key)

```
node ListAccessGroups.js
```

Sample output:

```
List access groups visible to the currently configured private key

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)

address                                    name
0x8d8780cfa0970a064e247e4a7829f0106b38d7f7 dev-tenant Content Admins
0xde9742280f18724d7b6b2d8bc7f3d96b538bc265 dev-tenant Content Users
0xce2a975ee73a8091cc7d451ed7c74a1de2739617 dev-tenant Tenant Admins


Done.
```

ListFiles.js

- Lists files in an Object or a Version of an Object
- Lists file path within Object, size, whether the file was stored encrypted, and (for S3 links) the remote path
- Note that S3 links always show `false` for 'encrypted', as they are not actually stored in the Content Fabric
- If you wish to list files in an old Version of an Object, use `--versionHash` instead of `--objectId`.
- NOTE: output will not include rows for directories that contain no files

```
node ListFiles.js \
  --objectId $EXISTING_OBJECT_ID
```

Sample output:

```
Get file list for iq__ho5NsUpev3AQv4XY1WvN8AwBXDz

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3

path                       size      encrypted link_remote_path
/audio-English.mp4         11832588  true
/audio-French.m4a          34906041  false     /MyS3Bucket/audio-French.m4a
/subtitles-English.vtt     10050     false     /MyS3Bucket/subtitles-English.vtt
/video-1080p.mp4           692307468 false     /MyS3Bucket/video-1080p.mp4
/images/Poster.png         465358    false


Done.
```

ListLibraries.js

- List libraries visible to the current user (i.e. the current private key)
- By default, shows only object IDs. Add `--name` to retrieve library names as well

```
node ListLibraries.js \
  --name
```

Sample output:

```
Get list of libraries

Getting list of libraries...
Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Found 4 lib(s)

libraryId                         name
ilib2nLKiR5p2yiGqCNicszxQYyvu9W4 dev-tenant - Properties
ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3 dev-tenant - Title Masters
ilib29dvmbN91uyXRwcMX88CAs8q2zeT dev-tenant - Title Mezzanines
ilib3MrFBvGaJeL5rCJeMg2KGPZsVecH dev-tenant - Reports


Done.
```

ListParts.js

- Lists parts and their sizes for an object (either for the latest version or an older version)
- If you specify only `--objectId` the newest version of the object will be used
- If you specify `--versionHash` then that specific version will be used

```
node ListParts.js \
  --objectId $MEZ_OBJECT_ID
```

Sample output:

```
Get part list for iq__ksXLW12ZAAg7PcLS4gKLUPPRUin

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT

hash                                                   size
hqpeCZ1A5NMuDfy7cfiNyH6EcVLCK2BSQ3pjvKBW5paXpD7jao8b   925761
hqpet8ye86mZMNy5ezhNRvCsU1kbFnPmcHw94LvJUQfHNA2mzSvNb  35218275
hqpeLisD3pdrNic2DdAjz5Urh8ErXzof5jBfQjW7dsSgzkvWfxBS   939009
hqpe72iLCJExTcHJXPqGJBdChby172ojSuxyKM5u9uaeL2U4cfFyh  34556322
hqpePowsTMpCH5emtLUN7Uf8V1KuTcinJgpbRKYMm1DRp5aEA8gG   937281
hqpez5TEN6YTPGF76LGz6UEaehf3bKDszDXvwxs1B65WHjRc9XWby  35600739
hqpeBrDpWx7cYJEfHKYR1s4Mq8sEBz2QdrgpGLGT3LZKZNVetQqN   940161
hqpeR5vrfeXjc7MGGtXuHqbLfbWYud1Hi8kHxUsHjUghUkRaTU1E   938433
hqpe2vL1CqJ16NoQjZmhH7RvP2D7NsTjYnLU3HfF24s8q4Mu4Ev53M 35137059
hqpeLssL5hBxfz9TdhXdimVDsnweDbaSX4REstQ1uQfizpNAgPK6   937281
hqpeH3g6wZfDbJsGANWkxhgL3d3JQ7ucj3Bjccp7zXN23qkfqa96   936129
hqpeS2o6TdSiwgA3pV2FFyiZKSS3YmzacXnAzYK86XMukW7bji2C   934977
hqpeioeUYZawMo4NNFpyuG6xBXpTeN5BtzKeA1Ys2UquK3wBfbL5E  35743587
hqpeRNTcnFQFsC4WhqXrRPpoaPLL8UoB8hgZvjX5262DZh5i38De   935553
hqpeFE17VkMC7tMZSWs68J7gughtALsRcLUtw3ffM4df2piWq1PE   934977
hqpeJHuRpcNdgf8yTSQFJrdbBcEZ2ZuoYzTqu3vHyr1Hev95iDBrY  36212004
hqpeMTkVDt8f7oSTf8fuWz1LGx5LCeYf8Q1iH9siWaDSGvRAhrN4   936129
hqpe2dqxXbNDNVwayzW7rM53mD8b3iRipPiBuER3v7yMZcV9bEXcVB 30266334
hqpeD4XQzj9R75hFuWy7M1tMJT7nUnyMs2pt5fF3Ge5oXd9f1sNG   936129
hqpe2Hk7Gk43gBjfvefFrXWmVhN2jXDyur94c2sy1asA9T2VBKTC7v 28922973
hqpeFAi8kbqG8UiGpNpsGKer8snLU9LundrBe1PD16FjJBvmMdBmG  31473759
hqpeR2VZdJwuHwsfAUv7ZNfKgzYSkum4KqQX1XP6TFNzJ9SStCv4   935553
hqpeK6YCnLgq4miikU3AE44Fz3E9KWXqrVmxx9VroUfzhmVEWisy   667137
hqpe24PLwF79cqhjYWR2AZMVbYrxFSLJhndtvyeVYAry9ZZyzBR4ow 31188063
hqpedMzBN45ATinUHQasv5NuKaTusZqnjyUNHFEozw9x9gXCotu5Y  32440992
hqpecEFqxRhJcCjv9Tsa3FG7Fsfj3ZDFgaDGyc9jk2j9tNS3mstZA  11664396
hqpe4md3QF83QnrvLKCoMSLko9hwKUDwzS346oedp36mZHg4L2bC   1756482
hqpe2MKUjgTkiH2NckUH6v6v8qVjB96sojKxUN6BuS9JNcoN2R69Mw 13623630
hqpe2w36KTfwUfDoWv3TKjsppyYyqKW2aQSx1Jo9JYdpms62XKpMZ5 25695258
hqpekHqXPcT3wysw3iLGdy9j8y6uZknPD6AUaQGXHuxnMyGX99VkM  5904198


Done.
```

ListTypes.js

- Lists content types visible to the current user (i.e. the current private key)

```
node ListTypes.js
```

Sample output:

```
Get list of content types

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)

id                                name
iq__3j9xhdqvzkGvzk5cGSBsD4mk12TG dev-tenant - Channel
iq__3168aeRL4ve6YAE4HX1huEVW6E6T dev-tenant - Eluvio LIVE Drop Event Site
iq__MeiZ3F266sx1hk3QvccVsLhQKfp  dev-tenant - Eluvio LIVE Marketplace
iq__bdQBYaURdXCiHZNkH78B56wQ6eJ  dev-tenant - Eluvio LIVE Tenant
iq__4U84NCVvoVH51249dFDobj2ZfZrK dev-tenant - Live Stream
iq__2jh5w9vsnE16P2tbd2xNiazcYsbZ dev-tenant - NFT Collection
iq__3oGYU8DaVfjsu4GiyALzVvnBcYfU dev-tenant - NFT Template
iq__4WQZhWEPmRT8rq5DLSJxBMnRhBAk dev-tenant - Permissions
iq__8SLzhEyJWiJ41BPezhswG56MUwL  dev-tenant - Title
iq__TSDVBqGVstjJzhYBXuWTKE2AZaw  dev-tenant - Title Collection
iq__2tfLjovW8zMN9Yh6eLmwynX1Cbip dev-tenant - Title Master


Done.
```

ListVersions.js

- Lists versions for a fabric object, newest first.
- By default, shows only version hashes, but options are available to show additional fields (use `--help` for a full list)

```
node ListVersions.js \
  --objectId $MEZ_OBJECT_ID \
  --commitMsg
```

Sample output:

```
List versions for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving version list for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Retrieving metadata path '/commit' from version
       hq__DWDY2vfAYGb44VKNuYZhsMNoxmnqQLixf4BY3qYxD7mdijGdwnBkqA9j5AR6k2fb7v5QawaVoY...
Retrieving metadata path '/commit' from version
       hq__CzMYsQ35s8BsWdhQX5LKf6CnyMgUjJj4TbNQHCpgQbX3SDh7Zp5jkqk95KAREZN62TYNtEbyyL...
Retrieving metadata path '/commit' from version
       hq__KjTSXUBer3zPKCy81nM6o2shmQTM9EqKJR17MxQMb2j76sdUaSrbvtVMCKrBw3iajBpcFHMGPk...

hash                                                                               commit_message
hq__DWDY2vfAYGb44VKNuYZhsMNoxmnqQLixf4BY3qYxD7mdijGdwnBkqA9j5AR6k2fb7v5QawaVoY Finalize ABR
       mezzanine
hq__CzMYsQ35s8BsWdhQX5LKf6CnyMgUjJj4TbNQHCpgQbX3SDh7Zp5jkqk95KAREZN62TYNtEbyyL Mezzanine LRO
       status
hq__KjTSXUBer3zPKCy81nM6o2shmQTM9EqKJR17MxQMb2j76sdUaSrbvtVMCKrBw3iajBpcFHMGPk Create ABR
       mezzanine

Done.
```

LROStatus.js

- Gets detailed information for a *single* LRO (*Long-Running Object*, i.e. a server-side process)
- This is a low-level script for debugging and development purposes - to get ingest status, use `utilities/MezzanineJobStatus.js` instead.
- Requires a write token (starts with `"tqw__"`) and an LRO Id (starts with `"tlro"`)
- LROs run on the node that generated the write token. If you know the write token's node you should use `--nodeUrl` to supply that node's URL as well, this will allow the script to skip lookup and considerably speed up execution. (e.g. `--nodeUrl "https://host-154-14-185-98.contentfabric.io"`)

```
node LROStatus.js \
  --writeToken  $WRITE_TOKEN \
  --lroId       $LRO_ID \
  --nodeUrl     $NODE_URL
```

Sample output:

```
Get status for LRO: tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUnCyXbifQnft6HYYWhCWtMi (write token:
       tqw__HSQWSzYbwughHe5JjRexFhNmLJeEGUcticW82cnDXv3n7hd7rXa2tcL2N6HtKinLJpv8YzTm3PYCMyqNTih)

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__DLVBv1Mcq68cPE1H662udqvXYyZ...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Checking that write token exists...
{
  handle: 'tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUnCyXbifQnft6HYYWhCWtMi',
  external_id:
       'audio.default.xcAudio.Video.mp4.tqw__HSQWSzYbwughHe5JjRexFhNmLJeEGUcticW82cnDXv3n7hd7rXa2
       tcL2N6HtKinLJpv8YzTm3PYCMyqNTih',
  description: 'stream: audio, offering: default, variant: default, masterHash:
       hq__HsEzV7tWekPDptg1oYNruYB92kgVwSaetMoiUq4CcTequJuRWnTYb9wWMub59tyHBReta5q5GN',
  state: 'terminated',
  history: [
    { state: 'registered', at: '2023-01-17T18:56:59.121Z' },
    {
      state: 'running',
      at: '2023-01-17T18:56:59.121Z',
      after: '85.226µs'
    },
    {
      state: 'terminating',
      at: '2023-01-17T18:56:59.726Z',
      after: '605.652389ms'
    },
    {
      state: 'terminated',
      at: '2023-01-17T18:56:59.726Z',
      after: '605.681234ms'
    }
  ],
  custom: {
    start: '2023-01-17T18:56:59Z',
    end: '2023-01-17T18:56:59Z',
    duration: '604ms',
    duration_ms: 604,
    run_state: 'finished',
    progress: { percentage: 100 }
  }
}

Done.
```

LROStop.js

- Sends a termination signal to a *single* LRO (*Long-Running Object*, i.e. a server-side process)
- This is a low-level script for debugging and development purposes - to stop a running ingest, use `utilities/MezJobStop.js` instead.
- Requires a write token (starts with `"tqw__"`) and an LRO Id (starts with `"tlro"`)
- After the `LROStop.js` script finishes, a transcode LRO may still take some time to completely shut down (e.g. for x265 / very high bitrate transcodes or ones involving a slow/distant S3 server). The LRO will, however, update the Mezzanine draft's status immediately, so `MezJobStatus.js` script will show that the LRO is cancelled without having to wait for it to finish shutting down.
- After terminating, `LROStatus.js` will show that LRO run state has changed to: `"cancelled by user"`
- LROs run on the node that generated the corresponding write token. If you know the write token's node you should use `--nodeUrl` to supply that node's URL as well, this will allow the script to skip lookup and considerably speed up execution. (e.g. `--nodeUrl "https://host-154-14-185-98.contentfabric.io"`)

```
node LROStop.js \
  --writeToken  $WRITE_TOKEN \
  --lroId       $LRO_ID \
  --nodeUrl     $NODE_URL
```

Sample output:

```
Stop LRO: tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUnCyXbifQnft6HYYWhCWtMi (write token:
       tqw__HSQWSzYbwughHe5JjRexFhNmLJeEGUcticW82cnDXv3n7hd7rXa2tcL2N6HtKinLJpv8YzTm3PYCMyqNTih)

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2DyMUGHtM7dsmfnvA3mhbypPqbXC...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT

Done.
```

MasterAddVariant.js

- Add a new variant to a production master.
- Use `--streams` to define contents of the variant. You can specify either a JSON string (converted to make it safe for the command line) or a path to a JSON file.

```
# using path to a JSON file (recommended):

node MasterAddVariant.js \
  --objectId         $MASTER_OBJECT_ID \
  --variantKey       french \
  --streams          ../example-files/variant_streams.json
```

```
# using inline JSON:

node MasterAddVariant.js \
  --objectId         $MASTER_OBJECT_ID \
  --variantKey       french \
  --streams '{"audio_fr":{"default_for_media_type":true, "label":"Français",
      "language":"fr-fr", "sources":[{"files_api_path":"audio-french.mp4",
      "stream_index":0}], "type":"audio"},
      "video":{"default_for_media_type":true, "label":"video", "language":"",
      "mapping_info":"", "sources":[{"files_api_path":"video-1080p.mp4",
      "stream_index":0}], "type":"video"}}'
```

Sample output:

```
Add variant 'french' to production master: iq__ho5NsUpev3AQv4XY1WvN8AwBXDz

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Retrieving metadata path '/production_master' from object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Saving changes...
Getting write token...
New write token:
       tqw__HSRvTnR3BpmgffhUQgNyc9KeJUA8AGLLbUqAPDcqNunKTbhf4oVrmqYnxmaV423cS1tbNuRZAwyBZxiMRze
       node URL: http://localhost:8008/
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__BdhKMSNXZHqKxfWzXGYEHZVtaQmxt2m1viET4XVUepdpVLbahQJyfPYmMaWBz6w7iFiBtdjGvk
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__BdhKMSNXZHqKxfWzXGYEHZVtaQmxt2m1viET4XVUepdpVLbahQJyfPYmMaWBz6w7iFiBtdjGvk

Done.
```

Example JSON file contents:

```
{
  "audio_fr": {
    "default_for_media_type": true,
    "label": "Français",
    "language": "fr-fr",
    "sources": [
      {
        "files_api_path": "audio-french.mp4",
        "stream_index": 0
      }
    ],
    "type": "audio"
  },
  "video": {
    "default_for_media_type": true,
    "label": "video",
    "language": "",
    "mapping_info": "",
    "sources": [
      {
        "files_api_path": "video-1080p.mp4",
        "stream_index": 0
      }
    ],
    "type": "video"
  }
}
```

MasterCopyVariant.js

- Copy an existing variant in a production master to a new variant key

```
node MasterCopyVariant.js \
  --objectId $MASTER_OBJECT_ID \
  --variantKey default \
  --newVariantKey french
```

Sample output:

```
Copy variant 'default' to 'french' in production master: iq__ho5NsUpev3AQv4XY1WvN8AwBXDz

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Retrieving metadata path '/production_master/variants' from object
        iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Saving changes...
Getting write token...
New write token:
        tqw__HSRvTnR3BpmgffhUQgNyc9KeJUA8AGLLbUqAPDcqNunKTbhf4oVrmqYnxmaV3vSLSFKKGUxQ1QihXaZaCX6
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__3RaVh24vdJGeiS1aGYc82zMCQdMfdg7J5iUHHdJTuCZjygB8SUMKuTmWeij3q9BUcUuY5oywa8
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__3RaVh24vdJGeiS1aGYc82zMCQdMfdg7J5iUHHdJTuCZjygB8SUMKuTmWeij3q9BUcUuY5oywa8

Done.
```

MasterCreate.js

- Creates a new production master object using the specified file(s)
- Adds files to the new object, probes them for media streams, searches for a video and audio stream to use, then automatically creates a default variant using those streams.
- If you upload multiple files, the automatic stream selection may not produce what you want. You can use `--streams` to explicitly specify what stream(s) from what file(s) to use (see the `MasterAddVariant.js` script for example usage).

```
node MasterCreate.js \
  --libraryId $MASTER_LIB \
  --type $MASTER_TYPE \
  --title "my video" \
  --files Video.mp4
```

Sample output:

```
Create production master

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up content type: iq__2tfLjovW8zMN9Yh6eLmwynX1Cbip...
Video.mp4: 0.0%
Video.mp4: 100.0%

Production master object created:
  Object ID: iq__389ZvHKvvfx1ns8XBTyVFc7PLBSx
  Version Hash: hq__68vCS5JQvZ9eGLryXhoxaYZHBCbC7Tz1d3AfQHiAQSpJDN29pNP28SqCtygPBJjWKFpEyqFsrG


Done.
```

MasterDeleteVariant.js

- Remove an existing variant from a production master.

```
node MasterDeleteVariant.js \
  --objectId $MASTER_OBJECT_ID \
  --variantKey french
```

Sample output:

```
Delete variant 'french' from production master: iq__ho5NsUpev3AQv4XY1WvN8AwBXDz

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Retrieving metadata path '/production_master/variants' from object
       iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Saving changes...
Getting write token...
New write token:
       tqw__HSRvTnR3BpmgffhUQgNyc9KeJUA8AGLLbUqAPDcqNunKTbhf4oVrmqYnxmaV42LiS5eJYJvEsGxMZRuiKvB
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__Ab6wRLTfHxZfc41ctUgBzyBEBj6EwNAKeGyr4TsZgd4aLAWTnVBSoCqRm34Z7D5v6ubY8j1pac
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__Ab6wRLTfHxZfc41ctUgBzyBEBj6EwNAKeGyr4TsZgd4aLAWTnVBSoCqRm34Z7D5v6ubY8j1pac

Done.
```

MasterInfo.js

MasterMakeDefaultVariant.js

- Creates the `default` Variant in a Master.
- Does not delete other Variants.
- Will exit with an error if the `default` Variant already exists.
- Media files must already exist in the Master, and must have already been scanned (e.g. with `MasterUpdateSources.js`).
- Will output a warning if only 1 stream is found (i.e. only video or only audio)
- Will exit with error if no suitable streams are found.

```
node MasterMakeDefaultVariant.js \
  --objectId $EXISTING_OBJECT_ID
```

Sample output:

```
Create default variant in master object iq__3Ag5bYenWyGepdKxnfvWax6VWVqt

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__3Ag5bYenWyGepdKxnfvWax6VWVqt...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Getting write token...
New write token:
        tqw__HSW95G48UbpabzPtdMjuXo7maTLGiXX3rYLMzS6n4wWg5EaaHT9h85Ji5YmFTBnjNuxtabLX4tZoyjxrukB
       node URL: http://localhost:8008
Finalizing object...
Finalized, new version hash:
        hq__LQDPyaBEt8jN8DH4MnmLaKRiLMXJ7EyrJkWQq83Sp1BzJjMp1YYA4Nt1rZWfGpNNfYJzDR9LXt
Waiting for publishing to finish and new object version to become available...
New object version now available
Retrieving metadata path '/production_master/variants' from version
        hq__LQDPyaBEt8jN8DH4MnmLaKRiLMXJ7EyrJkWQq83Sp1BzJjMp1YYA4Nt1rZWfGpNNfYJzDR9LXt...

New version hash: hq__LQDPyaBEt8jN8DH4MnmLaKRiLMXJ7EyrJkWQq83Sp1BzJjMp1YYA4Nt1rZWfGpNNfYJzDR9LXt


Done.
```

MasterUpdateSources.js

- Probes top-level files in object to find audio/video, then updates metadata at `/production_master/sources`.
- Used after a master has new or revised files added.
- Will warn if any variants point to non-existent files/streams after update.
- Use `--files` to choose what files to probe (if omitted, all top-level files will be probed).

```
node MasterUpdateSources.js \
  --objectId $MASTER_OBJECT_ID
```

Sample output:

```
Probe media files in master and update sources metadata for object ID:
       iq__3Ag5bYenWyGepdKxnfvWax6VWVqt

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__3Ag5bYenWyGepdKxnfvWax6VWVqt...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Retrieving metadata path '/production_master' from object iq__3Ag5bYenWyGepdKxnfvWax6VWVqt...
Probing BBB_480p_stereo_90s.mp4...
Probing Video.mp4...
Probing audio-part-filenames.txt...
Warnings:
failed to generate media.Source from media.Probe for file 'audio-part-filenames.txt':
       NewStreamFromGeneric: unrecognized format name: tty

Checking streams in variant 'default'...
  Checking stream 'audio'...
  Checking stream 'video'...

Files probed: BBB_480p_stereo_90s.mp4, Video.mp4, audio-part-filenames.txt
Files succeeded: BBB_480p_stereo_90s.mp4, Video.mp4

Sources added: BBB_480p_stereo_90s.mp4
Sources removed:
Sources with changed media info:

Saving changes...
Getting write token...
New write token:
       tqw__HSW95G48UbpabzPtdMjuXo7maTLGiXX3rYLMzS6n4wWg5EaaHT9h85Ji5YmFTU9V8HHpwu1DqPSwsvvbiy7
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__75jdKYTd9jdeV2LgAVgsDG8kLexjNchT4W8UNMgsyWyjmJt4CF8V7y1osJ7YFmGgRMYXEYuHFG
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__75jdKYTd9jdeV2LgAVgsDG8kLexjNchT4W8UNMgsyWyjmJt4CF8V7y1osJ7YFmGgRMYXEYuHFG

Done.
```

MetaCopy.js

- Copies metadata from one location (path) to another within an Object
- The metadata paths (`--path` and `--targetPath`) must start with `/`
- If `--targetPath` already exists in the Object's metadata, you must use `--force` to overwrite the existing value (even if the current value is `null`)
- If either metadata path contains a number (e.g. `/offerings/3/ready`) it will be interpreted as either a field name or an array index depending on whether an array currently exists at the corresponding location in the metadata
- If the target path contains field names / array indexes that do not exist, they will be created if possible. For example, if your current metadata is `{"baz": 42}` and you specify

  `--path /baz --targetPath /foo/bar`

  then after running the command the metadata will be `{"baz": 42, "foo": {"bar": 42}}`
- If it is not possible to create all the needed field names / array indexes, the script will fail and output an error message explaining why.

```
node MetaCopy.js \
  --objectId $EXISTING_OBJECT_ID \
  --path /offerings/default \
  --targetPath /offerings/bonus
```

Sample output:

```
Copy metadata for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin from /offerings/default to
       /offerings/bonus

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving metadata from object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Getting write token...
New write token:
       tqw__HSS5L3hAfyhWXfkDZtisrdCx9cc1M2xXh3FVT4pLBPX2h8jaa58Zi2Ap8bZLREyXRCy7xcBf97YWo1oVUcm
      node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__HQRT8fKyiQ3Y1dBSoN1ww2GXW7zSUpDm3nAjz2EBxEVPJP2Aoqd7svGgHUnrLLyvPsqUTtGn3L
Waiting for publishing to finish and new object version to become available...
New object version now available

Done.
```

MetaDelete.js

- Deletes metadata from a location (path) within an Object
- The metadata paths (`--path`) must start with `/`
- If the metadata path contains a number (e.g. `/offerings/3/ready`) it will be interpreted as either a field name or an array index depending on whether an array currently exists at the corresponding location in the metadata
- If the metadata path does not exist the script will fail and output an error message.

```
node MetaDelete.js \
  --objectId $EXISTING_OBJECT_ID \
  --path /offerings/bonus
```

Sample output:

```
Delete metadata path '/offerings/bonus' from object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving metadata from object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Getting write token...
New write token:
        tqw__HSS5L3hAfyhWXfkDZtisrdCx9cc1M2xXh3FVT4pLBPX2h8jaa58Zi2Ap8bZLRXxuAz6RC8ApLUzPsrYT9oq
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__iEdUWJA8UFevoPwofVFDCqJmMXd76LJrJEKVhquUm5WyoDJdFY3ku3eZRN3Q3ne2Y6aKJgqir
Waiting for publishing to finish and new object version to become available...
New object version now available

Done.
```

MetaGet.js

- Retrieves metadata from an Object
- Use `--path` if you wish to retrieve only a specific part of the metadata. (If omitted, all metadata visible to your private key will be returned)
- If the metadata path contains a number (e.g. `/offerings/3/ready`) it will be interpreted as either a field name or an array index depending on whether an array currently exists at the corresponding location in the metadata
- If the metadata path does not exist the script will fail and output an error message.
- Use `--outfile $FILEPATH_TO_SAVE_TO` if you wish to save the metadata to a file.
- If you wish to retrieve metadata from an old Version of an Object, use `--versionHash` instead of `--objectId`.

```
node MetaGet.js \
  --objectId $EXISTING_OBJECT_ID \
  --path /public/name
```

Sample output:

```
Get metadata for iq__ksXLW12ZAAg7PcLS4gKLUPPRUin path: /public/name

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Decoding object id from hash
        hq__iEdUWJA8UFevoPwofVFDCqJmMXd76LJrJEKVhquUm5WyoDJdFY3ku3eZRN3Q3ne2Y6aKJgqir...
Found: iq__ksXLW12ZAAg7PcLS4gKLUPPRUin
Reading metadata from version
        hq__iEdUWJA8UFevoPwofVFDCqJmMXd76LJrJEKVhquUm5WyoDJdFY3ku3eZRN3Q3ne2Y6aKJgqir...
Retrieving metadata path '/public/name' from version
        hq__iEdUWJA8UFevoPwofVFDCqJmMXd76LJrJEKVhquUm5WyoDJdFY3ku3eZRN3Q3ne2Y6aKJgqir...

"My test object name"

Done.
```

MetaMove.js

- Moves metadata from one location (path) to another within an Object
- The metadata paths (`--path` and `--targetPath`) must start with `/`
- If `--targetPath` already exists in the Object's metadata, you must use `--force` to overwrite the existing value (even if the current value is `null`)
- If either metadata path contains a number (e.g. `/offerings/3/ready`) it will be interpreted as either a field name or an array index depending on whether an array currently exists at the corresponding location in the metadata
- If the target path contains field names / array indexes that do not exist, they will be created if possible. For example, if your current metadata is `{"baz": 42}` and you specify

  `--path /baz --targetPath /foo/bar`

  then after running the command the metadata will be `{"foo": {"bar": 42}}`
- If it is not possible to create all the needed field names / array indexes, the script will fail and output an error message explaining why.

```
node MetaMove.js \
  --objectId $EXISTING_OBJECT_ID \
  --path /offerings/default \
  --targetPath /offerings/bonus
```

Sample output:

```
Move metadata for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin from /offerings/default to
       /offerings/bonus

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving metadata from object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Getting write token...
New write token:
       tqw__HSS5L3hAfyhWXfkDZtisrdCx9cc1M2xXh3FVT4pLBPX2h8jaa58Zi2Ap8bZLRBAYBsPQdg1U11on1SKEnPH
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__BysXtXJe7dTaPmzi1ZGWC1FU2rEzck1ak5w97Z9R5YjMJjvBSkTBbGXDLmPwztktTfLyyHkFrS
Waiting for publishing to finish and new object version to become available...
New object version now available

Done.
```

MetaSet.js

- Sets value at specific location (path) in the metadata of an Object
- The metadata path (`--path`) must start with `/`
- If the path already exists in the Object's metadata, you must use `--force` to overwrite the existing value (even if the current value is `null`)
- Use `--metadata` to specify the value you wish to store at the path. You can specify either a JSON string (converted to make it safe for the command line) or a path to a JSON file.
- If the metadata path contains a number (e.g. `/offerings/3/ready`) it will be interpreted as either a field name or an array index depending on whether an array currently exists at the corresponding location in the metadata
- If the path contains field names / array indexes that do not exist, they will be created if possible. For example, if your current metadata is empty (i.e. just `{}`) and you specify

  `--path /foo/bar --metadata 42`

  then after running the command the metadata will be `{"foo": {"bar": 42}}`
- If it is not possible to create all the needed field names / array indexes, the script will fail and output an error message explaining why.

```
# using path to a JSON file (recommended):

node MetaSet.js \
  --objectId $EXISTING_OBJECT_ID \
  --path /public/name \
  --force \
  --metadata ../example-files/metadata_simple_text.json
```

```
# using a JSON string:
#  original JSON = "My new title"
#  TextConvertForCmd.js used to convert to '"My new title"'

node MetaSet.js \
  --objectId $EXISTING_OBJECT_ID \
  --path /public/name \
  --force \
  --metadata '"My new title"'
```

Sample output:

```
Set metadata at /public/name for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin

Reading file /elv-utils-js/example-files/metadata_simple_text.json...
Initializing elv-client-js...  (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving existing metadata from object...
Retrieving metadata from object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Data already exists at '/public/name', --force specified, replacing...
Overwritten data: "My test object name…
```

```
Getting write token...
New write token:
        tqw__HSS5L3hAfyhWXfkDZtisrdCx9cc1M2xXh3FVT4pLBPX2h8jaa58Zi2Ap8bZLRV3fcMQbMtbay3a6cr5xbfp
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__8Px4kN4DL4GkR5VAMdY759N3VoFiEL51ye1xsNMVGHwk4NDaFLZ2sHUF3RADMLRN73ttWVYKXQ
Waiting for publishing to finish and new object version to become available...
New object version now available

Done.
```

MezCopyOffering.js

- Copies an existing Offering to a new Offering Key in the same Mezzanine Object.

```
node MezCopyOffering.js \
  --objectId $MEZ_OBJECT_ID \
  --offeringKey $EXISTING_OFFERING_KEY \
  --newOfferingKey $NEW_OFFERING_KEY
```

Sample output:

```
Copy offering 'default' to 'backup_copy' in mezzanine: iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving metadata path '/offerings' from object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Saving changes...
Getting write token...
New write token:
        tqw__HSTuqcei5VXGHfH7kALX4a5tkdrarDXhxvcaqgHSwJRVN7BVuSoaviuMWNWgaQcYA1f8CpEVjBApYJ3bdrV
       node URL: http://localhost:8008/
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__Ja9uA7h84YLdYho7ZNd2rebwP6coYoDT63Y1Q9pJGSDa6X8ZDYv6oLacYEMQALAJEVhQ315KgC
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__Ja9uA7h84YLdYho7ZNd2rebwP6coYoDT63Y1Q9pJGSDa6X8ZDYv6oLacYEMQALAJEVhQ315KgC

Done.
```

MezCreate.js

- Starts a new Mezzanine Job to prepare (transcode) an Offering, either creating a new Mezzanine Object as part of the process or reusing an existing Mezzanine Object.
- If `--objectId` is supplied *(or any of the aliases that `MezCreate.js` accepts for it, i.e. `--object-id`, `--existingMezId`, or `--existing-mez-id`)*, the existing Mezzanine Object will be reused. Otherwise a new Mezzanine Object will be created.
- Use `--variantKey` to choose which Variant in the Master to use as the basis for the Offering. If not specified, the script will look for a Variant named `"default"`
- If you do not want to include all streams in the Variant, use `--streamKeys` followed by a list of the streams you wish to include.
- Use `--offeringKey` to specify the Offering Key (i.e. the name to store the Offering under). If not specified, `"default"` will be used.
- You can request creation of additional modified copies of your Offering with `--addlOfferings`. You can specify either a JSON string (converted to make it safe for the command line) or a path to a JSON file.
  - Uses JSON Patch to specify modifications.
  - See `/elv-utils-js/example-files/addl_offering_specs.json` for an example.
  - If you use `--addlOfferings` you must also specify `--abrProfile`
- If an existing Mezzanine Object is being reused, then `--type` and `--title` can be omitted.
- If an existing Mezzanine Object is being reused, and an Offering with the same Offering Key already exists, then normally the entire pre-existing Offering will be replaced. If instead you would like to merge the stream(s) you are about to transcode into the existing Offering, add `--keepOtherStreams` and only old streams with the same Stream Key will be replaced *(see Replace a stream (revised master file provided after initial ingest) for an example).*
- If the file is short and you don't mind tying up the command line during transcoding, use `--wait` to automatically monitor progress and finalize the Mezzanine Job after it finishes.
- For most ingests you will not use `--wait` , and instead use `MezJobStatus.js` to check progress and finalize. *(You will need the Object ID output by the `MezCreate.js` script)*

```
node MezCreate.js \
  --libraryId $MEZ_LIB \
  --type $MEZ_TYPE \
  --title $NAME_FOR_OBJECT \
  --masterHash $MASTER_VERSION_HASH \
  --abrProfile $PATH_TO_YOUR_ABR_PROFILE
```

Sample output:

```
Create Mezzanine offering 'default' from variant 'default' in Master version
      hq__Gb9KgaM9FPpQzCR8Y7DRe9Qnv7VS5mBiWWwKQpi3T3WrDunWsv1x9zTz4gnSwGSNRJC2VpBFQt

Reading file /elv-utils-js/example-files/abr_profile_both.json...
Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up content type: iq__8SLzhEyJWiJ41BPezhswG56MUwL...
Creating new mezzanine object...
Starting Mezzanine Job(s)

Library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
```

```
Object ID: iq__47YQsxesRViBtpwzndxB6pxm2Dbp
Offering: default
Write Token:
       tqw__HSYsCLmDi4UP1wu5JdTMFwiS8eG1nkweUkVVoaX5b24KZnndpcdtSxKPwF51R7KSUqY8GrYcUyFsz9VukPP
Write Node: http://localhost:8008/


Done.
```

MezDeleteOffering.js

- Deletes an existing Offering from a Mezzanine Object.

```
node MezDeleteOffering.js \
  --objectId $MEZ_OBJECT_ID \
  --offeringKey $EXISTING_OFFERING_KEY
```

Sample output:

```
Delete offering 'backup_copy' from mezzanine: iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving metadata path '/offerings' from object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Saving changes...
Getting write token...
New write token:
        tqw__HSTuqcei5VXGHfH7kALX4a5tkdrarDXhxvcaqgHSwJRVN7BVuSoaviuMWNWgaqQucXbdaezKZ58A6Yopk4r
       node URL: http://localhost:8008/
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__9iPUgD3ibXtss67zTgAmB3qV5cuf3MxJZBGVpDJGTjBwso8oTnLaGTzsXDDuADfro6fr2coGji
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__9iPUgD3ibXtss67zTgAmB3qV5cuf3MxJZBGVpDJGTjBwso8oTnLaGTzsXDDuADfro6fr2coGji

Done.
```

MezJobCancel.js

- Terminates any running transcodes for a mezzanine.
- Exists with an error if no cancellable transcodes are found
- Cancelled transcodes will show run state `"cancelled by user"`

```
node MezJobCancel.js \
  --objectId $MEZ_OBJECT_ID
```

Sample output:

```
Cancel mezzanine job transcode(s)...

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__YtksuAbSELAM7NGLaddxTpAtMLS...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Found 2 transcode(s):

  stream: audio, offering: default, variant: default, masterHash:
       hq__8VF3n9rUUNA4PS3uXf8946Rxt5LX5cjzrvpvsGDZDe4x33P2AA3k8Jt4XMgdxTJKi8QJNADBs2
  LRO id: tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUVtPksxVwSFYHY6izCywhDd
  run state 'running'

  stream: video, offering: default, variant: default, masterHash:
       hq__8VF3n9rUUNA4PS3uXf8946Rxt5LX5cjzrvpvsGDZDe4x33P2AA3k8Jt4XMgdxTJKi8QJNADBs2
  LRO id: tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUMF7q5FBCMymyLE58Jiy7vv
  run state 'running'

Found 2 cancellable transcode(s), sending cancel request to each of the following:

  stream: audio, offering: default, variant: default, masterHash:
       hq__8VF3n9rUUNA4PS3uXf8946Rxt5LX5cjzrvpvsGDZDe4x33P2AA3k8Jt4XMgdxTJKi8QJNADBs2:
    run_state: 'running'

  stream: video, offering: default, variant: default, masterHash:
       hq__8VF3n9rUUNA4PS3uXf8946Rxt5LX5cjzrvpvsGDZDe4x33P2AA3k8Jt4XMgdxTJKi8QJNADBs2:
    run_state: 'running'

Waiting for cancel confirmation...

    LRO tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUVtPksxVwSFYHY6izCywhDd still shows as 'running'
        (stream: audio)
    LRO tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUMF7q5FBCMymyLE58Jiy7vv still shows as 'running'
        (stream: video)
    Waiting 15 seconds...
    All LRO(s) that were requested to cancel now report run state: 'cancelled by user'

Done.
```

MezJobStatus.js

- Gets status of transcode(s) for a mezzanine.
- If all transcodes finished successfully, you can also finalize the job with `--finalize`.
- If any of the transcodes finished with a progress percentage other than 100%, you can attempt to force finalization anyway with `--finalize --force`.

```
node MezJobStatus.js \
  --objectId $MEZ_OBJECT_ID
```

Sample output:

```
Get status for mezzanine job(s)

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__47YQsxesRViBtpwzndxB6pxm2Dbp...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
{
  "LROs": {
    "tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUShuiSfS8y7HqNjbjW4UB6J": {
      "desc": "stream: audio, offering: default, variant: default, masterHash:
        hq__Gb9KgaM9FPpQzCR8Y7DRe9Qnv7VS5mBiWWwKQpi3T3WrDunWsv1x9zTz4gnSwGSNRJC2VpBFQt",
      "duration": 108161000000,
      "duration_ms": 108161,
      "key":
        "audio.default.xcAudio.Meridian-video-1080p-audio-English-stereo.mp4.tqw__HSYsCLmDi4UP1wu5
        JdTMFwiS8eG1nkweUkVVoaX5b24KZnndpcdtSxKPwF51R7KSUqY8GrYcUyFsz9VukPP",
      "name": "MEZMAKER AUDIO (stream key: audio)",
      "progress": {
        "percentage": 16.666666666666664
      },
      "run_state": "running",
      "start": "2023-02-12T03:49:09Z",
      "seconds_since_last_update": 6,
      "estimated_time_left_seconds": 535,
      "estimated_time_left_h_m_s": "8m 55s",
      "eta_local": "7:59:57 PM PST"
    },
    "tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUsfuyC3wvRytfxnxoMJSk8u": {
      "desc": "stream: video, offering: default, variant: default, masterHash:
        hq__Gb9KgaM9FPpQzCR8Y7DRe9Qnv7VS5mBiWWwKQpi3T3WrDunWsv1x9zTz4gnSwGSNRJC2VpBFQt",
      "duration": 94732000000,
      "duration_ms": 94732,
      "key":
        "video.default.xcVideo.Meridian-video-1080p-audio-English-stereo.mp4.tqw__HSYsCLmDi4UP1wu5
        JdTMFwiS8eG1nkweUkVVoaX5b24KZnndpcdtSxKPwF51R7KSUqY8GrYcUyFsz9VukPP",
      "name": "MEZMAKER VIDEO (stream key: video)",
      "progress": {
        "percentage": 12.5
      },
      "run_state": "running",
      "start": "2023-02-12T03:49:09Z",
      "seconds_since_last_update": 19,
      "estimated_time_left_seconds": 644,
      "estimated_time_left_h_m_s": "10m 44s",
      "eta_local": "8:01:46 PM PST"
    }
  },
  "summary": {
    "run_state": "running",
    "estimated_time_left_seconds": 644,
    "estimated_time_left_h_m_s": "10m 44s",
    "eta_local": "8:01:46 PM PST"
  }
}

Done.
```

Finalizing:

```
node MezJobStatus.js \
  --objectId $MEZ_OBJECT_ID \
  --finalize
```

Sample output (finalizing):

```
Get status for mezzanine job(s)

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__47YQsxesRViBtpwzndxB6pxm2Dbp...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
{
  "LROs": {
    "tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUShuiSfS8y7HqNjbjW4UB6J": {
      "desc": "stream: audio, offering: default, variant: default, masterHash:
        hq__Gb9KgaM9FPpQzCR8Y7DRe9Qnv7VS5mBiWWwKQpi3T3WrDunWsv1x9zTz4gnSwGSNRJC2VpBFQt",
      "duration": 475562000000,
      "duration_ms": 475562,
      "end": "2023-02-12T03:57:05Z",
      "key":
        "audio.default.xcAudio.Meridian-video-1080p-audio-English-stereo.mp4.tqw__HSYsCLmDi4UP1wu5
        JdTMFwiS8eG1nkweUkVVoaX5b24KZnndpcdtSxKPwF51R7KSUqY8GrYcUyFsz9VukPP",
      "name": "MEZMAKER AUDIO (stream key: audio)",
      "progress": {
        "percentage": 100
      },
      "run_state": "finished",
      "start": "2023-02-12T03:49:09Z"
    },
    "tlro1EjchYzPaMx7VJV3JM8EmxaSfnyK4UDUsfuyC3wvRytfxnxoMJSk8u": {
      "desc": "stream: video, offering: default, variant: default, masterHash:
        hq__Gb9KgaM9FPpQzCR8Y7DRe9Qnv7VS5mBiWWwKQpi3T3WrDunWsv1x9zTz4gnSwGSNRJC2VpBFQt",
      "duration": 479663000000,
      "duration_ms": 479663,
      "end": "2023-02-12T03:57:09Z",
      "key":
        "video.default.xcVideo.Meridian-video-1080p-audio-English-stereo.mp4.tqw__HSYsCLmDi4UP1wu5
        JdTMFwiS8eG1nkweUkVVoaX5b24KZnndpcdtSxKPwF51R7KSUqY8GrYcUyFsz9VukPP",
      "name": "MEZMAKER VIDEO (stream key: video)",
      "progress": {
        "percentage": 100
      },
      "run_state": "finished",
      "start": "2023-02-12T03:49:09Z"
    }
  },
  "summary": {
    "run_state": "finished"
  }
}
Finalizing mezzanine...

ABR mezzanine object finalized:
  Object ID: iq__47YQsxesRViBtpwzndxB6pxm2Dbp
  Version Hash: hq__2zSTbHmDGPVVvxrzFndzwkwJtSu9YgGVuE1V4HxrATgwS596oX5XXd1fV2hxzrpKjwEq2yWJ3U

Waiting for publishing to finish and new object version to become available...
  new version not available yet, waiting 15 seconds...
New object version now available

Done.
```

MezRegenDrmKeys.js

- Used to update old mezzanines ingested prior to the 2023-02 Media Ingest Enhancements update. (Offerings created post-update generate a full set of DRM keys by default, unless an ABR Profile with `"store_clear": true` is used)
- Creates a new, full set of DRM keys for all audio/video streams in all offerings in a mezzanine object (except for offerings that were ingested with `"store_clear": true` - such offerings cannot have DRM added)
- Allows adding any DRM playout format to old offerings that did not have that format specified during initial ingest (as long as the offering is configured with `"store_clear": false`)
- Removes orphaned DRM keys from metadata at `/elv/crypt/drm/kids` and `/offerings/OFFERING_KEY/playout/drm_keys`
- If an offering has `"audio_individual_drm_keys": true` configured, then each audio stream in that offering will receive its own set of DRM keys. Otherwise, all audio streams in an offering will share the same set of keys (metadata path: `/offerings/OFFERING_KEY/audio_individual_drm_keys`)

```
node MezRegenDrmKeys.js \
  --objectId $MEZ_OBJECT_ID
```

Sample output:

```
Regenerate DRM keys for all offerings in object iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving existing metadata from object...
Retrieving metadata from object iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ...
Getting write token...
New write token:
        tqw__HSTmDD5NSjfdZNv7AEyiUWq3eacZRBaM5FxJyA6CCUDpMdwyQfiWYhDiUujrukSN7aU1fkxfUvCqSD4TsAg
        node URL: http://localhost:8008
  Processing offering 'clear'...
    Offering 'clear' has "store_clear": true, skipping...
  Processing offering 'default'...
Finalizing object...
Finalized, new version hash:
        hq__AaKnupFkaMM5MEDUtyK3XqRFqc7vK3sV6A7t54rzuQtCizgQVLtjsFR8cHNBipxyLFG2QxvcNG
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__AaKnupFkaMM5MEDUtyK3XqRFqc7vK3sV6A7t54rzuQtCizgQVLtjsFR8cHNBipxyLFG2QxvcNG

Done.
```

MezSetCodecDescs.js

- Used to update old mezzanines ingested prior to the 2023-02 Media Ingest Enhancements update. (Offerings created pre-update advertise incorrect video levels for some rungs in HLS/DASH playlists)
- Sets the codec descriptor string for all video rungs in a mezzanine's offering(s)
- Fixes playout issues encountered with some hardware players
- Use `--offeringKey` to fix a single offering, otherwise all offerings in the object will be processed.

```
node MezSetCodecDescs.js \
  --objectId $MEZ_OBJECT_ID
```

Sample output:

```
Set codec descriptor strings in bitrate ladder for video stream(s) in object
       iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ.

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving metadata path '/offerings' from object iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ...
Processing offering 'clear'...
  Retrieving playout/clear/dash-clear/video/videovideo_1920x1080@9500000/init.m4s...
    codec desc: avc1.640028
  Retrieving playout/clear/dash-clear/video/videovideo_1280x720@4500000/init.m4s...
    codec desc: avc1.64001f
  Retrieving playout/clear/dash-clear/video/videovideo_960x540@2000000/init.m4s...
    codec desc: avc1.64001f
  Retrieving playout/clear/dash-clear/video/videovideo_768x432@1100000/init.m4s...
    codec desc: avc1.640016
  Retrieving playout/clear/dash-clear/video/videovideo_640x360@810000/init.m4s...
    codec desc: avc1.42c016
  Retrieving playout/clear/dash-clear/video/videovideo_640x360@520000/init.m4s...
    codec desc: avc1.42c016
Processing offering 'default'...
  Retrieving playout/default/dash-clear/video/videovideo_1920x1080@9500000/init.m4s...
    codec desc: avc1.640028
  Retrieving playout/default/dash-clear/video/videovideo_1280x720@4500000/init.m4s...
    codec desc: avc1.64001f
  Retrieving playout/default/dash-clear/video/videovideo_960x540@2000000/init.m4s...
    codec desc: avc1.64001f
  Retrieving playout/default/dash-clear/video/videovideo_768x432@1100000/init.m4s...
    codec desc: avc1.640016
  Retrieving playout/default/dash-clear/video/videovideo_640x360@810000/init.m4s...
    codec desc: avc1.42c016
  Retrieving playout/default/dash-clear/video/videovideo_640x360@520000/init.m4s...
    codec desc: avc1.42c016
Getting write token...
New write token:
       tqw__HSTmDD5NSjfdZNv7AEyiUWq3eacZRBaM5FxJyA6CCUDpMdwyQfiWYhDiUujruP6GEg6pJeGEoaCshTAYihY
      node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__GABFnPj1HN1QBJZN5KEsAGYwLRWBSFfaUSVoM7mMFH2Y7EyBJxDSoppbuTZqJRonaTFGSVG1YQ
Waiting for publishing to finish and new object version to become available...
New object version now available

Done.
```

MezUnifyAudioDrmKeys.js

- Used to update old mezzanines ingested prior to the 2023-02 Media Ingest Enhancements update. (Works around a Bitmovin player issue when switching audio streams. Offerings created post-update have their audio streams share DRM keys by default unless an ABR Profile with `"audio_individual_drm_keys": true` is used)
- Sets all audio streams in each Offering of a Mezzanine to use the same set of DRM keys, by copying the keys for one audio stream of an Offering to all other audio streams in the same Offering.
- Also sets `"audio_individual_drm_keys": false` for all Offerings, so that if `MezRegenDrmKeys.js` script is run, all audio streams in an Offering will continue to share DRM keys.

```
node MezUnifyAudioDrmKeys.js \
  --objectId $EXISTING_MEZ_OBJECT_ID
```

Sample output:

```
Edit Mezzanine iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA to use 1 shared set of audio DRM keys per
       Offering

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving existing metadata from object...
Retrieving metadata from object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
  Checking offering 'default'...
    Using keys from stream 'audio'...
Getting write token...
New write token:
       tqw__HSTuqcei5VXGHfH7kALX4a5tkdrarDXhxvcaqgHSwJRVN7BVuSoaviuMWNWgavSxWMXjKaRuhz2yezkJPXC
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__4peicpLgWvqnnJdQJCoW54wfMkxu9V34uj3WcVtZPVpAAnTUPwjY8oW8aW6RxD4jDVAcnd8xxY
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__4peicpLgWvqnnJdQJCoW54wfMkxu9V34uj3WcVtZPVpAAnTUPwjY8oW8aW6RxD4jDVAcnd8xxY

Done.
```

ObjectAddGroupPerm.js

- **THIS PAGE (AND SCRIPT) MAY NEED REVISION**
- Grants specified permission for an Object to an Access Group
- There are three permission types:
    - `see` - Allows reading public metadata (stored in metadata path `/public`)
    - `access` - Allows reading entire object (both public and private data)
    - `manage` - Allows reading entire object (both public and private data) AND changing the object, including its permissions
- Note that the permissions are cumulative - if you grant `manage` permission to a group, you do not need to grant `access` or `see` permissions. *(If you do grant multiple permission types to a group this will not cause a problem, but if you wish to revoke permissions later you will need to do this individually for each type that you granted)*
- For Masters and Mezzanines, it is common to grant `manage` permission to your `Content Admins` group after the Objects are created.

```
node ObjectAddGroupPerm.js \
  --objectId $EXISTING_OBJECT_ID \
  --groupAddress $ACCESS_GROUP_ADDRESS \
  --permission $PERMISSION_TYPE
```

Sample output:

```
Add 'manage' permission to iq__ksXLW12ZAAg7PcLS4gKLUPPRUin for group
       0x8d8780cfa0970a064e247e4a7829f0106b38d7f7

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Adding permission: manage...

Done.
```

ObjectCreate.js

- Creates an Object in the specified Library
- If you wish to set a Content Type use `--type`
- If you wish to set the Object's Metadata, use `--metadata`. You can specify either a JSON string (converted to make it safe for the command line) or a path to a JSON file.

```
node ObjectCreate.js \
  --libraryId $EXISTING_LIBRARY_ID \
  --name $NAME_FOR_OBJECT
```

Sample output:

```
Create object 'my new object' in lib ilib29dvmbN91uyXRwcMX88CAs8q2zeT

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Creating new draft object in library ilib29dvmbN91uyXRwcMX88CAs8q2zeT...
New object ID: iq__ZJvRWRKTd4Xegb4ZpyqLHvryPHS
New write token:
        tqw__HSRW1TTEKaE9NiQoJt9LT7dp4oTg12XMQCLVVetMQQsCUUgeFDnCyHVRzWVLUNUFqw3kDwLe9qC5MTzoZsu
Finalizing object...
Finalized, new version hash:
        hq__2d7bDowvo7Ukwh4R3DgpjJHvSGC61jLc9yUYKnxFw1W4AYKqHt2RBYLYo4GozMa88Vy1KQrxYg
Waiting for publishing to finish and new object version to become available...
New object version now available
New object ID: iq__ZJvRWRKTd4Xegb4ZpyqLHvryPHS
version hash: hq__2d7bDowvo7Ukwh4R3DgpjJHvSGC61jLc9yUYKnxFw1W4AYKqHt2RBYLYo4GozMa88Vy1KQrxYg

Done.
```

ObjectDelete.js

- **WARNING: DELETED OBJECTS CANNOT BE RECOVERED!**
- Deletes an existing Object.

```
node MezDeleteObject.js \
  --objectId $EXISTING_OBJECT_ID
```

Sample output:

```
Delete object iq__3P8qdCiBSsEdB1eedMiJ4idgbxjc

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__3P8qdCiBSsEdB1eedMiJ4idgbxjc...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT

Done.
```

ObjectDeleteVersion.js

- **WARNING: DELETED VERSIONS CANNOT BE RECOVERED!**
- Removes one Version from an existing Object
- You may wish to run **`node ListVersions.js --commitMsg --commitTime`** first to get a list of all the Versions of an Object

```
node ObjectDeleteVersion.js \
  --versionHash $EXISTING_OBJECT_VERSION_HASH
```

Sample output:

```
Delete version hq__AWnwyGhrVkM8aSvgMgADHs5uWnUz6NmvFMvssm7eYwH1yZhYVYEL3f44S6AgcQA2gvVrKXttVn

Deleting object version
       hq__AWnwyGhrVkM8aSvgMgADHs5uWnUz6NmvFMvssm7eYwH1yZhYVYEL3f44S6AgcQA2gvVrKXttVn...
Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)

Done.
```

ObjectGetPermission.js

- Gets the current Object Permission setting for an Object.
- The possible permission settings are
  - `owner` - Owner Only
  - `editable` - Editable
  - `viewable` - Viewable
  - `listable` - Publicly Listable
  - `public` - Public

```
node ObjectGetPermission.js \
  --objectId $EXISTING_OBJECT_ID
```

Sample output:

```
Get object permission setting for iq__4JhZtSENmDu7hTq91LEufzDnDpj3

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__4JhZtSENmDu7hTq91LEufzDnDpj3...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Permission: owner

Done.
```

ObjectListGroupPerms.js

- **THIS PAGE (AND SCRIPT) MAY NEED REVISION**
- Lists the current Access Group permissions for an Object.

```
node ObjectListGroupPerms.js \
  --objectId $EXISTING_OBJECT_ID
```

Sample output:

```
List group permissions for object iq__nVABoeoiGikEo6sVJ1EjT6DBWdr

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__nVABoeoiGikEo6sVJ1EjT6DBWdr...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT

address                                    name                      see access manage
0x8d8780cfa0970a064e247e4a7829f0106b38d7f7 dev-tenant Content Admins       x      x


Done
```

ObjectPruneVersions.js

- **WARNING: DELETED VERSIONS CANNOT BE RECOVERED!**
- Removes one or more Versions from an existing Object, either starting from the oldest or the newest, depending on options chosen.
- You may wish to run `node ListVersions.js --commitMsg --commitTime` first to get a list of all the Versions of an Object
- Use `--keep` to specify how many remaining Versions you wish to have after deletion
- Use `--keepNew` if you wish to keep the newest Version(s) (i.e. delete starting with the OLDEST version)
- Use `--keepOld` if you wish to keep the oldest Version(s) (i.e. delete starting with the NEWEST version)

```
node ObjectPruneVersions.js \
  --objectId $EXISTING_OBJECT_ID \
  --keep 3 \
  --keepOld
```

Sample output:

```
Prune object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA, keeping 3 version(s)

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving version list for object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Object current version count = 8, deleting 5 newest version(s)...
Deleting object version
	hq__9iPUgD3ibXtss67zTgAmB3qV5cuf3MxJZBGVpDJGTjBwso8oTnLaGTzsXDDuADfro6fr2coGji...
Deleting object version
	hq__Ja9uA7h84YLdYho7ZNd2rebwP6coYoDT63Y1Q9pJGSDa6X8ZDYv6oLacYEMQALAJEVhQ315KgC...
Deleting object version
	hq__H4s9QErok499YxisxjabRc3PazcSPjCKVB8ZW3R61W6na9VEcZBWrKd531wcQK5EYgzoNEDw3n...
Deleting object version
	hq__AXBSG128vHtoHgqKDCgem86p8P7j6AUHGXitcQ42qpPsMp3XNmvBofmMRouf1rbbiG7e8vfpek...
Deleting object version
	hq__3YS4yMFEp1yyomT2zELPt16DdqTbNxcmosNh83cWChdX4j7H7vg84CSAxAsq2jqpuxyNyQ64rC...

Done.
```

ObjectRemoveGroupPerm.js

- **THIS PAGE (AND SCRIPT) MAY NEED REVISION**
- Removes specified permission for an Object from an Access Group
- There are three permission types:
  - `see` - Allows reading public metadata (stored in metadata path `/public`)
  - `access` - Allows reading entire object (both public and private data)
  - `manage` - Allows reading entire object (both public and private data) AND changing the object, including its permissions
- Permissions must be removed individually - if you have granted all 3 permission types to an Access Group, to revoke all 3 would require 3 commands.
- You can view current group permissions on an Object with the `ObjectListGroupPerms.js` script.

```
node ObjectRemoveGroupPerm.js \
  --objectId $EXISTING_OBJECT_ID \
  --groupAddress $ACCESS_GROUP_ADDRESS \
  --permission $PERMISSION_TYPE
```

Sample output:

```
Remove 'manage' permission for iq__ksXLW12ZAAg7PcLS4gKLUPPRUin from group
       0x8d8780cfa0970a064e247e4a7829f0106b38d7f7

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Removing permission: manage...

Done.
```

ObjectSetPermission.js

- **THIS PAGE NEEDS REVISION**
- Sets the Object Permission for one Object
- ***Note that the displayed Object Permission can be misleading - there is a second kind of permission (Object Group Permissions) that can grant additional access, even if the Object Permission is set to e.g. "Owner Only"***
- The possible permission levels are
  - `owner` - Owner Only
  - `editable` - Editable
  - `viewable` - Viewable
  - `listable` - Publicly Listable
  - `public` - Public
- For a Mezzanine Object, the most common permission setting is `viewable`
- Depending on what the old and new permission levels are, it is possible that the Version Hash of the Object will change (certain changes of permission require edits to Object metadata). The script will output whether or not a new Version was created.

```
node ObjectSetPermission.js \
  --objectId $EXISTING_OBJECT_ID \
  --permission $PERMISSION_LEVEL
```

Sample output: (version hash **changed**)

```
Set permission on iq__ksXLW12ZAAg7PcLS4gKLUPPRUin to viewable

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Previous version hash:
       hq__ErxfDQX4tExvstzb8URyx9WG2BYBrWkAWB66kvzH6Hw3pkQKZzN8PmL6AGAftLyktCcataFxSW
New version hash: hq__FH1ZgDNyqr2Hj9eFRNNMvSdKfSpU3x5dpPufo2xWbnDKqiXrp9XFV7jDfgoFaXTwTt7aDgVh1t

Done.
```

Sample output: (version hash **unchanged**)

```
Set permission on iq__ksXLW12ZAAg7PcLS4gKLUPPRUin to editable

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ksXLW12ZAAg7PcLS4gKLUPPRUin...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Version hash unchanged:
       hq__FH1ZgDNyqr2Hj9eFRNNMvSdKfSpU3x5dpPufo2xWbnDKqiXrp9XFV7jDfgoFaXTwTt7aDgVh1t

Done.
```

ObjectSetType.js

- Sets the **Content Type** of an Object
- Once you set the Content Type, there is no way to clear it
- The `--type` option accepts either an ID, a Version Hash, or the name of a Content Type (enclose in quotes if it contains spaces, e.g. `"dev-tenant - Title"`)

```
node ObjectSetType.js \
  --objectId $EXISTING_OBJECT_ID \
  --type $TYPE_ID_OR_HASH_OR_NAME
```

Sample output:

```
Set content type on iq__2PDzvA279deFCZe2SV1B6NQQkDE to 'dev-tenant - Title'

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2PDzvA279deFCZe2SV1B6NQQkDE...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Looking up content type: dev-tenant - Title...
New version hash: hq__G6nafFuvMmQ7SjRdeFNTBKe8QFHujHT3is4bS4a8R2HUh682BMpFQxDm43atCeeHC3prmFTeuE

Done.
```

OfferingAddSubtitles.js

- Adds a subtitle stream to an Offering
- The subtitles must be in a WebVTT format file on your machine
- Use `--language` to specify a language code for the stream (see Language codes and labels for a list). Many players will use a device's locale setting to choose what subtitle track should be the default.
- Use `--label` to set how the stream will be listed in player subtitle menus.
- If the subtitle stream should be the default, add `--isDefault`
- If the subtitles are
- You must choose a `--streamKey` for the new stream (e.g. `subtitles-en` or `subtitles-en-forced`). This will become part of the URL that is used to retrieve the subtitle stream.
-

```
node OfferingAddSubtitles.js \
  --objectId $EXISTING_MEZ_OBJECT_ID \
  --streamKey $STREAM_KEY \
  --language $LANGUAGE_CODE \
  --label $DISPLAY_LABEL \
  --file $PATH_TO_VTT_FILE
```

Sample output:

```
Add subtitle stream to offering 'default' in object iq__3BsFDHB4HDGNizwE1VvNbqm4Re65.

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__3BsFDHB4HDGNizwE1VvNbqm4Re65...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving metadata path '/offerings/default' from object iq__3BsFDHB4HDGNizwE1VvNbqm4Re65...
Getting write token...
New write token:
        tqw__HSWCWb2WatGRdUzXMFKii5asn59emvDY4t5V1LFCYCiCBUTUVzapDN4NSno6MRsGwzvWeedEGhd4xhuW5Eq
       node URL: http://localhost:8008
Subtitles uploaded as new part: hqpeA6XdMsQ6jss4cvcUjL1i4ghGPy845tqLaWe6dzxc4DsdRb
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__3Dj1qZhJLAREs6YpnSueJuoX49XbBgWcpkmMooZ42FHZrnMgarCtsFi7EU5HjYuvJckdJMvxnu
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__3Dj1qZhJLAREs6YpnSueJuoX49XbBgWcpkmMooZ42FHZrnMgarCtsFi7EU5HjYuvJckdJMvxnu

Done.
```

OfferingAddVideoRung.js

OfferingDeleteStream.js

- Deletes a stream from an existing Offering

```
node OfferingDeleteStrEam.js \
  --objectId $MEZ_OBJECT_ID \
  --offeringKey $OFFERING_KEY \
  --streamKey $STREAM_KEY
```

Sample output:

```
Delete stream 'audio' from offering 'default' in object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA.

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving metadata path '/offerings/default' from object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Getting write token...
New write token:
       tqw__HSTuqcei5VXGHfH7kALX4a5tkdrarDXhxvcaqgHSwJRVN7BVuSoaviuMWNWgarwRBMtt8ihPKGCrKF8VVad
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__LYBf1dwwiSzW4WnRAViQ5C9jNaFGL8EFHPWKVCZmFyZRnZXcUy9yNziiTykXiTMSZWTnJS4aRE
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__LYBf1dwwiSzW4WnRAViQ5C9jNaFGL8EFHPWKVCZmFyZRnZXcUy9yNziiTykXiTMSZWTnJS4aRE

Done.
```

OfferingDeleteVideoRung.js

OfferingEditStream.js

OfferingInfo.js

OfferingSetFormats.js

- Changes playout formats for an offering.
- Use `--formats` to select which format(s) to make available. Valid choices are:
  - dash-clear
  - dash-widevine
  - hls-aes128
  - hls-clear
  - hls-fairplay
  - hls-sample-aes
- Clear formats can only be used if the offering specifies `"drm_optional": true`
- DRM formats can only be used if the offering specifies `"store_clear": false` and if all required DRM keys are present (if not, use `utilities/MezRegenDrmKeys.js` first)
- Use `--offeringKey` to specify which offering to change (if omitted, offering `"default"` will be edited)

```
node OfferingSetFormats.js \
  --objectId $MEZ_OBJECT_ID \
  --formats hls-fairplay dash-widevine
```

Sample output:

```
Set playout formats for offering 'default' in object iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ.

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving metadata path '/elv/crypt/drm/kids' from object iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ...
Retrieving metadata path '/offerings/default' from object iq__2LqUikZeBsrC38dXHFnzJyHGNMPJ...
Getting write token...
New write token:
       tqw__HSTmDD5NSjfdZNv7AEyiUWq3eacZRBaM5FxJyA6CCUDpMdwyQfiWYhDiUujrupcFyy1w9Lew2L5S7QRfRWG
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__9vLJH8rZGnwcEVoXRhvCuimsVowozde2fBYazdnzexU1ZAZ6yy4g116J3A26QtAgvxkzLpHX2L
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__9vLJH8rZGnwcEVoXRhvCuimsVowozde2fBYazdnzexU1ZAZ6yy4g116J3A26QtAgvxkzLpHX2L

Done.
```

OfferingSetImageWatermark.js

- Adds an image watermark to an Offering
- The image must already exist as a file in the Mezzanine Object (use the **`FilesAdd.js`** script if needed)
- If you wish to clear the watermark, use **`--clear`** instead of **`--watermark`**.
- Use **`--offeringKey`** to specify which offering to change (if omitted, offering **`"default"`** will be edited)
- You cannot have both a text and an image watermark on the same Offering.
- Use **`--watermark`** to configure the path to the image file and watermark size/placement. You can specify either a JSON string (converted to make it safe for the command line) or a path to a JSON file.
- The JSON to specify the image watermark has the following fields:
  - **`align_h`** - *(text)* horizontal alignment (**`"left"`**, **`"center"`**, or **`"right"`**)
  - **`align_v`** - *(text)* vertical alignment (**`"top"`**, **`"middle"`**, or **`"bottom"`**)
  - **`image`** - *(text)* the file path within the Object to the image to use
  - **`margin_h`** - *(text)* the horizontal margin
    - controls how much space to leave between the watermark and the left or right edge of video frame, depending on setting of **`align_h`**
    - has no effect if **`align_h`** is set to **`"center"`**
    - expressed as a fraction of the video frame width, e.g. **`"1/20"`** would leave 1/20th the frame width between the watermark and the left or right edge of the video frame
  - **`margin_v`** - *(text)* the vertical margin
    - controls how much space to leave between the watermark and the top or bottom edge of video frame, depending on setting of **`align_v`**
    - has no effect if **`align_v`** is set to **`"middle"`**
    - expressed as a fraction of the video frame height, e.g. **`"1/10"`** would leave 1/10th the frame height between the watermark and the top or bottom edge of the video frame
  - **`target_video_height`** - *(integer)* the video height (in pixels) that the image was designed for (e.g. if the image is 108 pixels high and **`target_video_height`** is set to **`1080`**, then the watermark will be 1/10th the height of the video frame. If you use the same image but set **`target_video_height`** to **`540`**, then the watermark will be 1/5th the height of the video frame)
- The example file **`/elv-utils-js/example-files/image_watermark.json`** makes use of a semi-transparent image file **`/elv-utils-js/example-files/image_watermark.png`**. After adding the image file to a Mezzanine Object, you can add the example watermark with:

```
node OfferingSetImageWatermark.js \
  --objectId $MEZ_OBJECT_ID \
  --watermark ../example-files/image_watermark.json
```

Sample output:

```
Set image watermark for offering 'default' in object iq__nVABoeoiGikEo6sVJ1EjT6DBWdr

Reading file /elv-utils-js/example-files/image_watermark.json...
Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__nVABoeoiGikEo6sVJ1EjT6DBWdr...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving existing offering metadata from object...
Retrieving metadata path '/offerings/default' from object iq__nVABoeoiGikEo6sVJ1EjT6DBWdr...
Getting write token...
New write token:
        tqw__HSS9ytiepZtgW6WipALVh6stP1fx7pCvtPSBUoSURaAfDnpYaPg6dLGFFMW4qMoS2j95XuDW7E5nAHPuU4K
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__BJsFzvpbjhZSWgCPZCiQtHd18DaeGYvNhWWZtUPoXkaXSxU6NiJkE8WbXFKaeiGW8PimnmDQDi
Waiting for publishing to finish and new object version to become available...
New object version now available
New Version Hash: hq__BJsFzvpbjhZSWgCPZCiQtHd18DaeGYvNhWWZtUPoXkaXSxU6NiJkE8WbXFKaeiGW8PimnmDQDi

Done.
```

- Example command using inline JSON:

```
node OfferingSetImageWatermark.js \
  --objectId $MEZ_OBJECT_ID \
  --watermark '{"align_h":"right", "align_v":"bottom",
       "image":"/image_watermark.png", "margin_h":"1/20", "margin_v":"1/10",
       "target_video_height":1080}'
```

Example JSON file contents:

```
{
  "align_h": "right",
  "align_v": "bottom",
  "image": "/image_watermark.png",
  "margin_h": "1/20",
  "margin_v": "1/10",
  "target_video_height": 1080
}
```

OfferingSetTextWatermark.js

- Adds a text watermark to an Offering
- If you wish to clear the watermark, use `--clear` instead of `--watermark`.
- Use `--offeringKey` to specify which offering to change (if omitted, offering `"default"` will be edited)
- You cannot have both a text and an image watermark on the same Offering.
- Use `--watermark` to configure the text contents and the watermark size/placement. You can specify either a JSON string (converted to make it safe for the command line) or a path to a JSON file.
- The JSON to specify the image watermark has the following fields:
    - `font_color` - *(text)* color and optional opacity level for the text, e.g. `"white@0.25"` would result in fairly transparent white text (only 25% opaque). For a full list of available color names as well as alternate ways of specifying the color see the ffmpeg-utils color syntax documentation.
    - `font_relative_height` - *(number)* font height relative to video frame (e.g. `0.05` = 5% or 1/20th the video frame height)
    - `shadow` - *(true/false)* whether to draw a shadow behind the text
    - `shadow_color` - *(text)* color and optional opacity level for the shadow (uses the same color syntax as `font_color`)
    - `template` - *(text)* the text content of the watermark. This text can be customized by the use of substitution variables. If `template` contains any of the following they will be replaced by information determined at playout time:
        - `$USERNAME` - a user ID associated with playback (requires additional configuration of your playout link generator)
        - `$USEREMAIL` - an email address associated with playback (requires additional configuration of your playout link generator)
        - `$HASH` - the Version Hash of the Mezzanine Object
        - `$META(`*`path`*`)` - metadata retrieved from the specified path of the Mezzanine Object
    - `x` - *(text)* the horizontal position for watermark text to start ("0" = left edge of video frame). There are variables available to express the position in terms of text width and video frame width:
        - `w` = video frame width
        - `tw` = watermark text total width
        - `"w - (tw / 2)"` would center the text horizontally on the screen
    - `y` - *(text)* the vertical position for watermark text to start ("0" = top edge of video frame). There are variables available to express the position in terms of text height and video frame height:
        - `h` = video frame height
        - `lh` = watermark text single line height
        - `th` = watermark text total height
        - `"h - (th / 2)"` would center the text vertically on the screen
        - `"h - (lh * 4)"` would start the text at the fourth line from the bottom edge

- The example file `/elv-utils-js/example-files/text_watermark.json` uses the `$USERNAME` substitution variable. You can add the example watermark with:

```
node OfferingSetTextWatermark.js \
  --objectId $MEZ_OBJECT_ID \
  --watermark ../example-files/text_watermark.json
```

Sample output:

```
Set text watermark for offering 'default' in object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA

Reading file /elv-utils-js/example-files/text_watermark.json...
Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Found library ID: ilib29dvmbN91uyXRwcMX88CAs8q2zeT
Retrieving existing offering metadata from object...
Retrieving metadata path '/offerings/default' from object iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA...
Getting write token...
New write token:
        tqw__HSTuqcei5VXGHfH7kALX4a5tkdrarDXhxvcaqgHSwJRVN7BVuSoaviuMWNWgaPdxgftJqSXWnLihb36dXHw
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__C9uzR6WP88rMr2QKNSZLE69jeebZwaSQFt5pArL1eZ2r6mLv1VgeEZf3HSiFKEfsyhHTEXU5Mz
Waiting for publishing to finish and new object version to become available...
New object version now available
New Version Hash: hq__C9uzR6WP88rMr2QKNSZLE69jeebZwaSQFt5pArL1eZ2r6mLv1VgeEZf3HSiFKEfsyhHTEXU5Mz

Done.
```

- Example command using inline JSON:

```
node OfferingSetTextWatermark.js \
  --objectId $MEZ_OBJECT_ID \
  --watermark '{"font_color":"white@0.5", "font_relative_height":0.05,
      "shadow":true, "shadow_color":"black@0.5", "template":"PREPARED FOR
      $USERNAME - DO NOT DISTRIBUTE", "x":"(w-tw)/2", "y":"h-(4*lh)"}'
```

Example JSON file contents:

```
{
  "font_color": "white@0.5",
  "font_relative_height": 0.05,
  "shadow": true,
  "shadow_color": "black@0.5",
  "template": "PREPARED FOR $USERNAME - DO NOT DISTRIBUTE",
  "x": "(w-tw)/2",
  "y": "h-(4*lh)"
}
```

SimpleIngest.js

TextConvertForCmd.js

- Will prompt for one line of input, then print out a version of the input that is converted into a form that is safe for use on the command line
- Primary use is to convert JSON to something that can be used safely in commands
- To convert multi-line JSON, first use a minifier like https://codebeautify.org/jsonminifier to convert to a single line.

```
node TextConvertForCmd.js
```

Sample input/output:

```
Enter text to convert:
{"audio_fr":{"default_for_media_type":true,"label":"Français","language":"fr-fr","sources":[{"files_api_path":"audio-french.mp4","stream_index":0}],"type":"audio"},"video":{"default_for_media_type":true,"label":"video","language":"","mapping_info":"","sources":[{"files_api_path":"video-1080p.mp4","stream_index":0}],"type":"video"}}


Input received
--------------------------
{"audio_fr":{"default_for_media_type":true,"label":"Français","language":"fr-fr","sources":[{"files_api_path":"audio-french.mp4","stream_index":0}],"type":"audio"},"video":{"default_for_media_type":true,"label":"video","language":"","mapping_info":"","sources":[{"files_api_path":"video-1080p.mp4","stream_index":0}],"type":"video"}}


Converted for command line
--------------------------
'{"audio_fr":{"default_for_media_type":true,"label":"Français","language":"fr-fr","sources":[{"files_api_path":"audio-french.mp4","stream_index":0}],"type":"audio"},"video":{"default_for_media_type":true,"label":"video","language":"","mapping_info":"","sources":[{"files_api_path":"video-1080p.mp4","stream_index":0}],"type":"video"}}'
```

TextDecodeBase58.js

- Will prompt for one line of input, then print out a version of the input that is converted from Base58 encoding to hex
- Can be used to convert Content Fabric IDs to Addresses (remove prefix first, e.g. for Library ID of `ilib29dvmbN91uyXRwcMX88CAs8q2zeT`, leave out the `ilib`)

```
node TextDecodeBase58.js
```

Sample input/output:

```
Enter text to decode from Base58: 29dvmbN91uyXRwcMX88CAs8q2zeT

Input received
--------------------------
29dvmbN91uyXRwcMX88CAs8q2zeT

Decoded from Base58
(decoded length: 40 digits)
--------------------------
0x5277a14300da23c1912061e8d0a9c825e47c5e08
```

TypeGet.js

- Gets the Content Type for a Library, Object, Version, or Draft. (Note an item may not have a Content Type set)
- If you wish to get the Content Type of an old Version of an Object, use **`--versionHash`** instead of **`--objectId`**.
- If you wish to get the Content Type of a Draft, use **`--writeToken`** instead of **`--objectId`**. If you know the URL of the Node that generated the Write Token, use **`--nodeUrl`** to specify it for faster retrieval.
- If you wish to get the Content Type of a Library, use **`--libraryId`** instead of **`--objectId`**.

```
node TypeGet.js \
  --objectId $EXISTING_OBJECT_ID
```

Sample input/output:

```
Get content type for object iq__4JhZtSENmDu7hTq91LEufzDnDpj3

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__4JhZtSENmDu7hTq91LEufzDnDpj3...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Looking up content type:
hq__C8jPrmX5ccivtXYmY24DiCxAANVGCFBD3mLRqRvu7nfBKZ2FewuerW5wSrgsKaSyPBrvwJXRBN...
{
  "name": "dev-tenant - Title Master",
  "objectId": "iq__2tfLjovW8zMN9Yh6eLmwynX1Cbip",
  "versionHash": "hq__C8jPrmX5ccivtXYmY24DiCxAANVGCFBD3mLRqRvu7nfBKZ2FewuerW5wSrgsKaSyPBrvwJXRBN"
}

Done.
```

VariantABRProfile.js

VariantAddStream.js

- Add a new stream to an existing production master variant.

```
node VariantAddStream.js \
  --objectId $MASTER_OBJECT_ID \
  --streamKey audio_fr \
  --lang fr-fr \
  --file audio-french.mp4 \
  --label French \
  --streamIndex 0
```

Sample output:

```
Remove stream 'audio_fr' from variant 'default' of production master:
       iq__ho5NsUpev3AQv4XY1WvN8AwBXDz

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Retrieving metadata path '/production_master/variants' from object
       iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Saving changes...
Getting write token...
New write token:
       tqw__HSRvTnR3BpmgffhUQgNyc9KeJUA8AGLLbUqAPDcqNunKTbhf4oVrmqYnxmaV41UkkcMR26P6tqBZ8vyDyAf
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__2niK9aeVLTbACVQ1i3mMnjuLfnRssuxW9ZxMdYDvNJezgxFB4ECFN9GVgHe1cdVGSzunmt2xPi
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__2niK9aeVLTbACVQ1i3mMnjuLfnRssuxW9ZxMdYDvNJezgxFB4ECFN9GVgHe1cdVGSzunmt2xPi

Done.
```

VariantDeleteStream.js

- Removes an existing stream from an existing production master variant

```
node VariantDeleteStream.js \
  --objectId $MASTER_OBJECT_ID \
  --streamKey audio_fr
```

Sample output:

```
Delete stream 'audio_fr' from variant 'default' of production master:
       iq__ho5NsUpev3AQv4XY1WvN8AwBXDz

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Retrieving metadata path '/production_master/variants' from object
       iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Saving changes...
Getting write token...
New write token:
       tqw__HSRvTnR3BpmgffhUQgNyc9KeJUA8AGLLbUqAPDcqNunKTbhf4oVrmqYnxmaV41UkkcMR26P6tqBZ8vyDyAf
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
       hq__2niK9aeVLTbACVQ1i3mMnjuLfnRssuxW9ZxMdYDvNJezgxFB4ECFN9GVgHe1cdVGSzunmt2xPi
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__2niK9aeVLTbACVQ1i3mMnjuLfnRssuxW9ZxMdYDvNJezgxFB4ECFN9GVgHe1cdVGSzunmt2xPi

Done.
```

VariantEditStream.js

- Edit a stream specification in an existing production master variant
- Command options similar to utilities/VariantAddStream.js, but you can list only the options you want to change
- To clear `--mapping`, `--channelIndex`, `--language`, or `--multipliers`, use the corresponding options `--clearMapping`, `--clearChannelIndex`, `--clearLanguage`, or `--clearMultipliers`

```
node VariantEditStream.js \
  --objectId $MASTER_OBJECT_ID \
  --streamKey audio_fr \
  --label Français
```

Sample output:

```
Edit stream 'audio_fr' in variant 'default' of production master: iq__ho5NsUpev3AQv4XY1WvN8AwBXDz

Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
Looking up library ID for object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Found library ID: ilib3fm7YhNrmYBNsgNwFuUso1CRVFw3
Retrieving metadata path '/production_master' from object iq__ho5NsUpev3AQv4XY1WvN8AwBXDz...
Saving changes...
Getting write token...
New write token:
        tqw__HSRvTnR3BpmgffhUQgNyc9KeJUA8AGLLbUqAPDcqNunKTbhf4oVrmqYnxmaV46hkkQ6nJF57EegRorXt22U
       node URL: http://localhost:8008
Writing metadata to object...
Finalizing object...
Finalized, new version hash:
        hq__5EDd1eksVkS9vu2KCg4jjmGJu41GDcPAyLzyHiUdioxVTuwSgZYvZLXDA2FGmDjbpe6jdDz6KA
Waiting for publishing to finish and new object version to become available...
New object version now available
New version hash: hq__5EDd1eksVkS9vu2KCg4jjmGJu41GDcPAyLzyHiUdioxVTuwSgZYvZLXDA2FGmDjbpe6jdDz6KA

Done.
```

WriteTokenInfo.js

- Displays information for a write token
- Use `--nodeUrl` to also determine the URL of the node that generated the write token (it may take a minute)

```
node WriteTokenInfo.js \
  --writeToken $TOKEN \
  --nodeUrl
```

Sample output:

```
Print info for write token
       tqw__HSTuqcei5VXGHfH7kALX4a5tkdrarDXhxvcaqgHSwJRVN7BVuSoaviuMWNWgaqQucXbdaezKZ58A6Yopk4r

Getting node info for write token
       tqw__HSTuqcei5VXGHfH7kALX4a5tkdrarDXhxvcaqgHSwJRVN7BVuSoaviuMWNWgaqQucXbdaezKZ58A6Yopk4r..
       .
Initializing elv-client-js... (config URL: http://localhost:8008/config?qspace=dev&self)
{
  "tokenType": "tq__",
  "tokenId": "0xd58b27bc8a7714ab38c6a8ee6c300743",
  "objectId": "iq__2Pq7oDT7gxopf1mURnPJyM3cjEFA",
  "nodeId": "inodXnRMo5b4svum81wHZtvpDq9DtUf",
  "nodeUrl": "http://localhost:8008/"
}

Done.
```

Reference: elv-utils-js /example-files

ABR Profiles

'Clear' profiles

There are 2 versions of ABR Profile without DRM:

- File names ending in `'...store_unencrypted.json'` - use this version for offerings that you want to make accessible to the general public (without any login or authorization). **DRM cannot be added later to these offerings unless you reingest with a different ABR profile**. (Note that if you reingest with DRM, the media will no longer be accessible without login/authorization)
- File names ending in `'...store_encrypted.json'` - use this version for media that you want to restrict access to (requiring login/authorization). You CAN add DRM later to offerings ingested with these profiles.

Library

Metadata

Offering

Variant

Watermark

Reference: Configuration File syntax and processing

(work in progress: precedence, preset directives, configuration file merging, substitution variable handling)

Reference: Language codes and labels

Below is a table of codes and labels for some of the most commonly encountered languages. To look up a language not on this list, try the following sources:

- Andiamo
- W3Docs
- Wikipedia

Note that some references may capitalize letters appearing after a dash, e.g. "pt-BR" instead of "pt-br".

Language	**ISO 639-1 Code**	**Label**	**Note**
Arabic	ar	اَلْعَرَبِيَّةُ	
Bangla / Bengali	bn	বাংলা	
Chinese (Cantonese)	yue	廣東話	*(spoken) for audio streams only*
Chinese (Mandarin)	cmn	普通话	*(spoken) for audio streams only*
Chinese (Simplified)	zh-hans	简体中文	*(written) for text streams only*
Chinese (Traditional)	zh-hant	繁體中文	*(written) for text streams only*
Danish	da	Dansk	
Dutch	nl	Nederlands	
Dutch (Netherlands)	nl-nl	Nederlands	*specifically as spoken/written in the Netherlands*
English	en	English	
English	en-gb	English (United Kingdom)	*specifically as spoken/written in the United Kingdom*
Finnish	fi	Suomi	
French	fr	Français	
French (Canada)	fr-ca	Français (Canada)	*specifically as spoken/written in Canada*
French (Parisian)	fr-fr	Français	*specifically as spoken/written in France*
German	de	Deutsch	

Language	ISO 639-1 Code	Label	Note
German (Germany)	de-de	Deutsch (Deutschland)	*specifically as spoken/written in Germany*
Hebrew	he	עִבְרִית	
Hindi	hi	हिन्दी	
Indonesian	id	Bahasa Indonesia	
Italian	it	Italiano	
Japanese	ja	日本語	
Korean	ko	한국어	
Marathi	mr	मराठी	
Norwegian	no	Norsk	
Polish	pl	Polski	
Portuguese	pt	Português	
Portuguese (Brazil)	pt-br	Português (Brasil)	*specifically as spoken/written in Brazil*
Russian	ru	Русский	
Spanish (Castilian)	es	Español	*specifically as spoken/written in Spain*
Spanish (Latin America)	es-419	Español (Latinoamericano)	*specifically as spoken/written in Latin America*
Swedish	sv	Svenska	
Telugu	te	తెలుగు	
Thai	th	ภาษาไทย	
Turkish	tr	Türkçe	