Kaltura
open source video

Developer Training
Focus: API Based Applications

Agenda

- Keep track of Kaltura, the Knowledge Hubs
- The Kaltura REST API
- API Authentication & Security and Media Access Control
- Entries Management API
- Categories Management API
- Custom Metadata API (Profiles, Schemas, Index, Search)
- Bulk Upload and Content Migration
- The Multirequest API
- PlayManifest and Download URLs
- Dynamic Thumbnails and Thumbnails Assets Management
- Kaltura Server Notifications
- Kaltura Players Framework and Player JavaScript API
- Kaltura Player Plugins

Kaltura
open source video

Keeping Track

Technical Resources -

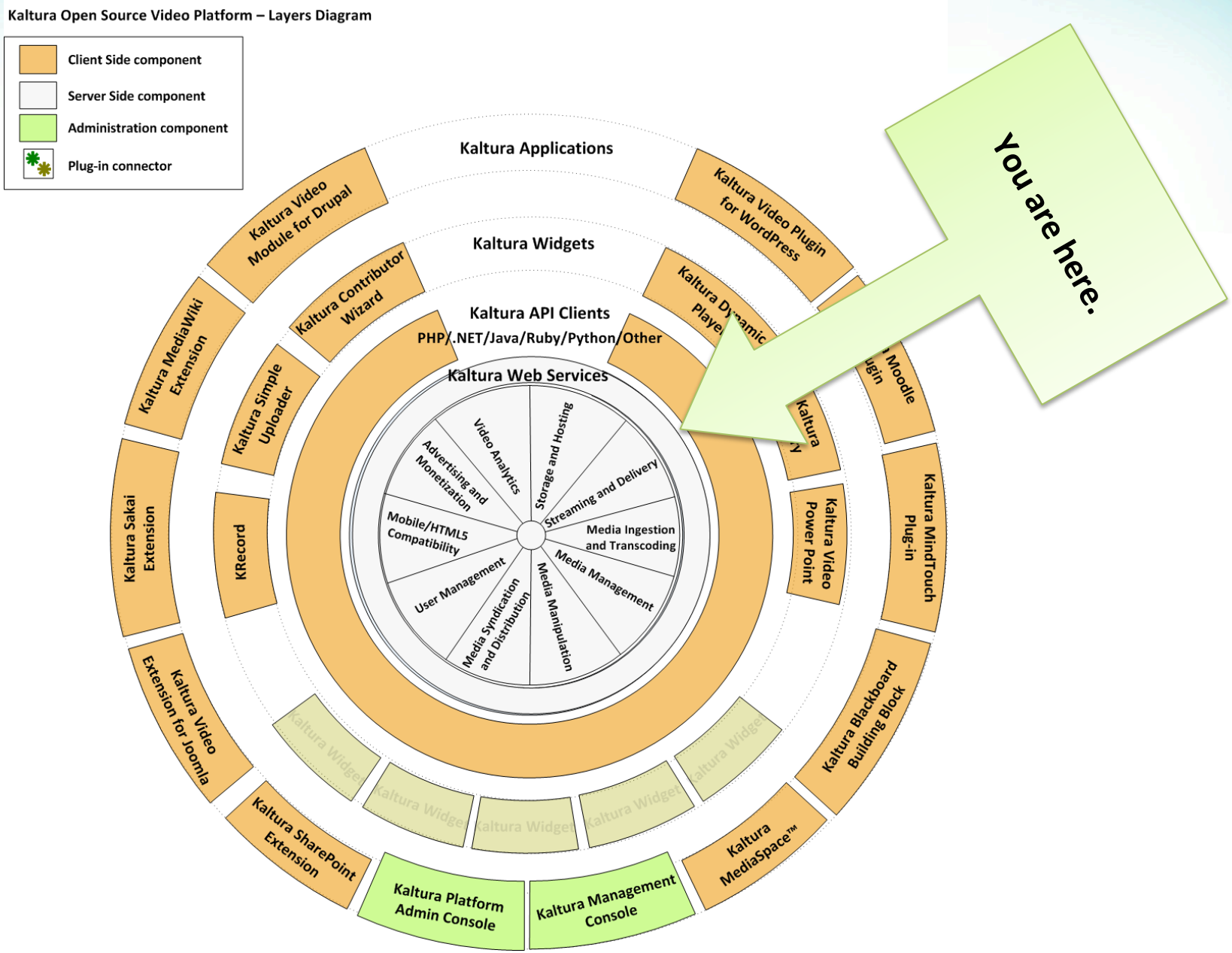
- http://knowledge.kaltura.com/kaltura-api-documentation-set
- http://blog.kaltura.org
- http://html5video.org and http://player.kaltura.com

Stay updated -

- http://videos.kaltura.com
- http://corp.kaltura.com/live-webinars
- http://exchange.kaltura.com
- The Kaltura Monthly Newsletter
- The Kaltura Facebook Group

Kaltura
open source video

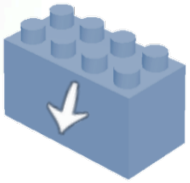
The Kaltura REST API



The Kaltura Video Platform Architecture Overview

Kaltura open source video

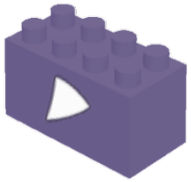
API For ALL!



Ingest
Upload and Bulk Upload, Import, Recording, Virus Scan

Manage
Asset Management, Metadata, Search, Transcoding , Thumbnails, Clip & Trim

Publish
Widgets, Playlists, CDN



Playback
VOD, Live, Adaptive, Cross-device

Distribute & Syndicate
Feeds, automatic content sync to 3rd party

Engage
Captions, Cue-Points, Annotations, Related, Ratings & Voting

Monetize
Ads, Ad Networks, Access Control, Paywalls

Analyze
Reports and Analytics

Administrate
Authentication, Session, Users, Roles & Permissions

Your API?
Being Open Source and Flexible Kaltura allows you to extend even it's core API!

Kaltura
open source video

The Kaltura REST API

- Controls every aspect of the Kaltura Platform
  - * No Tight-Coupling – If you see it, you can build it
- Based on "REST" principles – Simplicity & Flexibility
- XML schema - generated client libraries & test console
- State-less advanced authentication model
- Responses by default are returned as XML text. JSON, JSONP and PHP Serialized are also available.
- The API ignores Null fields, allowing efficient update actions

Introduction to the Kaltura API Architecture
Kaltura API Documentation Set

Kaltura
open source video

The Kaltura API Building Blocks

- The Kaltura API consists of ***Services and Actions*** for querying/adding/updating/listing entities and for activating processes within the Kaltura Backend.
- ***Actions*** are grouped in ***Services*** according to the entity type they apply to.

Introduction to the Kaltura API Architecture

Kaltura
open source video

Note About Backward Compatibility

- There were 3 major API releases in Kaltura.
  - You should always use the latest version when creating new applications.
- Kaltura is built to be backward as compatible as possible, often supporting APIs from the 1st version!
- Note that API actions marked as deprecated (in the Test Console or API Docs) will continue to be functional, even though not recommended.

Kaltura
open source video

Deliver Cross-Platform

Html5video.org
A Kaltura Community Site

Kaltura
open source video

The Kaltura API Client Libraries

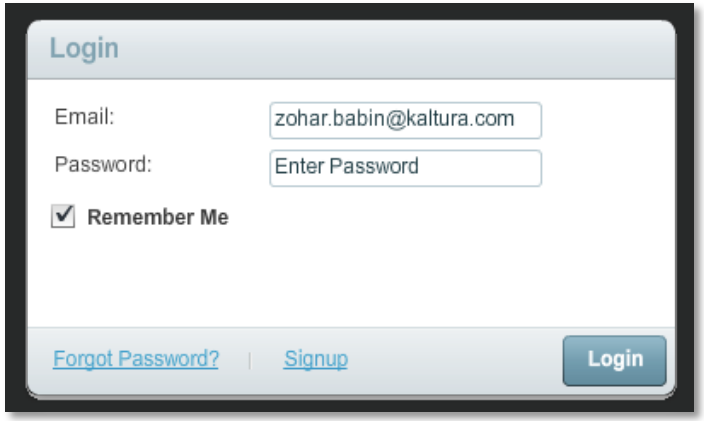
- Client libraries wrap the API in a familiar programming language (PHP, Java, Javascript, C#, etc…) and makes coding a lot easier!
- Without a client library, a call to an API action looks like this: `/api_v3/index.php?service=baseentry&action=list`
  With the PHP client library, it looks like this –
  `$results = $client->baseEntry->listAction();`
- Download the automatically generated client libs from: http://<host_name>/api_v3/testme/client-libs.php

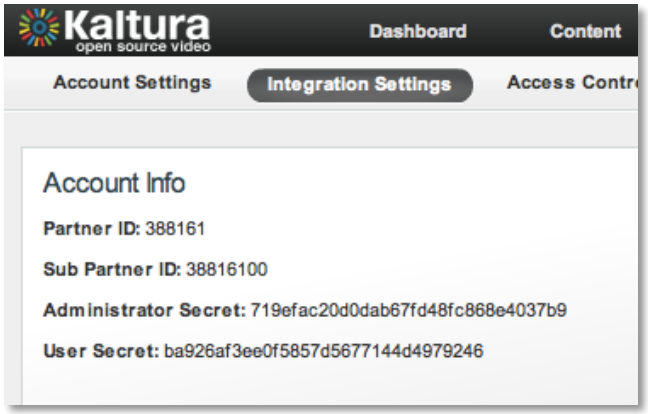
Introduction to Kaltura Client Libraries

Kaltura
open source video

Starting Point: Creating Kaltura Sessions

- The Kaltura Session is a string that identifies a valid API user session. It is commonly dubbed 'KS'
- To create a valid KS, you'll need; the account id ('partnerId'), user id and API keys or user credentials
- The Kaltura API is state-less:
  - You'll have to pass a valid KS in every consequent API request
  - Multiple KSs of the same user can be used simultaneously





How to Create a Kaltura Session

Kaltura
open source video

3 Methods To Create a Kaltura Session

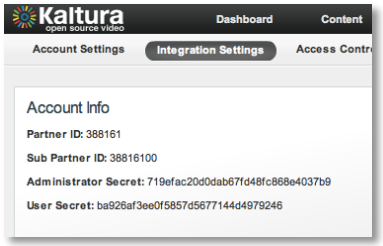
Method	Credentials Needed	Use-cases
session.start API call	• Publisher Account Id • Admin or User Secret API key	Only use with secured trusted applications that authenticate via HTTPS. Use when generating end-user sessions with special privileges
GenerateSession local client library function	• Publisher Account Id • Admin Secret API key	Only use with secured trusted applications. A local function saving a round-trip. This method assumes clock sync between Kaltura and your servers (NTP based) – for short time sessions best using session.start
user.loginByLoginId API call	• User email • User password	Use when users authenticate directly or with 3rd party applications.

[How to Create a Kaltura Session]

Kaltura
open source video

Authentication Considerations

- Users can be easily created, removed or blocked and their password can easily be changed.



- API Secret keys can never be changed.
- Sharing the account API secret keys with 3rd party vendors should be avoided, as secret keys can not be regenerated or blocked for access.
- Kaltura API based application developers and 3rd party application vendors should build their application to leverage the ***user.loginByLoginId*** API and ask the publisher for their email, password and account Id (aka partnerId).

[How to Create a Kaltura Session]

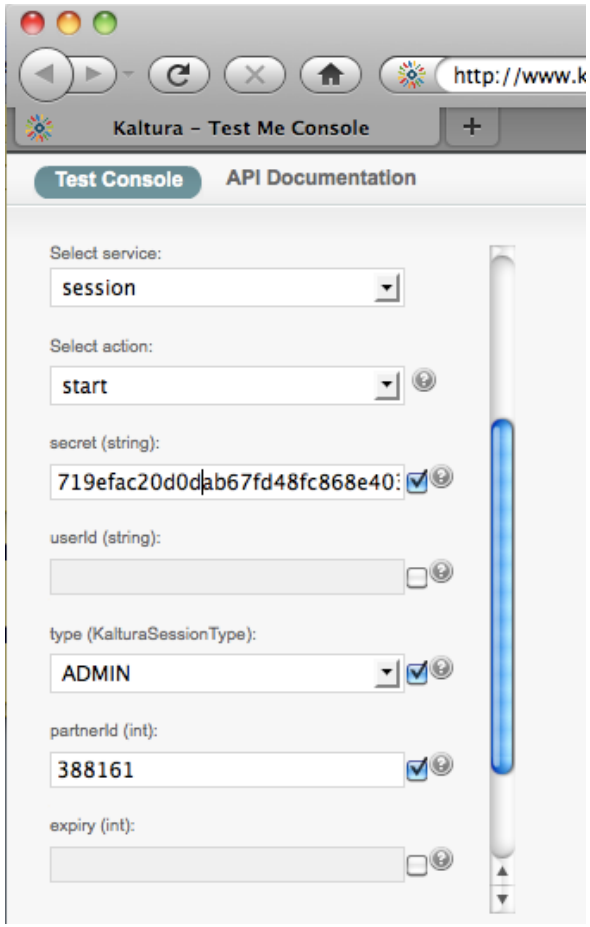
The Kaltura API Test Console

- Kaltura Servers also generates an API Test Console (aka "Kaltura TestMe").
- It can be accessed by browsing to: http://<host_name>/api_v3/testme
- Includes links to docs & client libraries download page
- Combine use with an HTTP Sniffer

Using Kaltura's API Test Console Introduction

Kaltura
open source video

Using the API Test Console – Let's create a Kaltura Session (aka KS)



[Using Kaltura's API Test Console Introduction]

Kaltura
open source video

After Creating a Kaltura Session (KS)

- The KS must be passed on all API actions
- All our actions called will be tracked as:
  - The **specific publisher account** specified in the KS
  - The **specific user** of that account specified in the KS
- Each action will be allowed or restricted according to the permission configurations of that user and publisher account and permissions set in the KS creation

Kaltura
open source video

Authentication and Security

Securing Kaltura Content - Background

Data -

- **The Kaltura Session.** Our focus today…
- The channel communication (e.g. HTTPS).

Delivery -

- Delivery obfuscation (tokenized urls).
- Secured delivery (RTMPE, HTTPS).
- SWF verification.
- DRM.

Kaltura's API Authentication and Security

Kaltura
open source video

Terminology and Objects

- **Kaltura Session (aka KS)**
  - The Kaltura Session holds the authentication session details, it must be provided with every API action call
- **Partner (aka Publisher)**
  - Represents a Kaltura account. partnerId is the global identifier of every account.
- **API Admin Secret Keys**
  - Secret key provided for every publisher account, two sets of keys: Admin and User.
- **KalturaUser**
  - Represent a person who uses the system.
  - Holds metadata information like name, email, etc...
  - 2 Types – admin / end-user.
  - May be associated to more then one partner.
- **Permission**
  - An object that represents a certain logical activity in the system.
  - Examples: Content Ingestion, Scheduling Management, etc…
  - A specific permission may have a different meaning for each application that uses it.
- **UserRole**
  - A set of permissions.
  - A user is associated with a role, which means that the user will “have” the permissions defined in that role.
  - The role can be different for each partner the user belongs to.

Kaltura
open source video

KS Type: User vs Admin

Admin	User
Admin by default (if no role or privileges were specifically assigned) can access all API actions	User by default (if no role or privileges were specifically assigned) can only access a subset of API actions, and these will only return entries he is privileged to
Can list & update all Entries	Can only access Entries owned by him, unless granted explicit privileges (like list:*)
Can set an Entry owner	Cannot modify ownership
Bypasses any Access Control restrictions	Abide all Access Control restrictions
When listing for LiveStream receives as response: KalturaLiveStreamAdminEntry	When listing for LiveStream receives as response: KalturaLiveStreamEntry

Kaltura
open source video

User Roles & Permissions

- Restrict access to specific API actions based on Roles.
- A Role, can be applied to
  - Users.
  - KS at run-time, to create a Role restricted KS.
- When a user has a defined Role, that will determine his set of allowed actions
- There is default role assigned to users that don't have a role (3 defaults - Widget, User, Admin – Configured via server side script)
- External applications like the KMC, admin console and MediaSpace, use Roles & Permissions in order to “act” differently according to a user’s permissions.

Kaltura
open source video

KMC Permissions - Example

- Roles can be created and applied to Admin users in the Administration panel
- Every permission is mapped to a group of UI elements and their attributes. Buttons, pages, modules.
  - The KMC can show/hide UI elements like tabs, buttons, etc. according to the user's permissions.

Kaltura
open source video

Media Access Control

- The access control API service accessControlProfile .
- Limits access to Entries based on predefined set of rules such as:
  - User Authentication – is user logged in?
  - Geolocation – Country codes
  - IP Address range
  - Metadata
  - Parent domain (where the player is embedded)
  - User Agent (e.g. limit Mobile access)
  - A valid KS was provided
- **Time limit (Scheduling)** - Allow access to media within a time period only. E.g., allow users to access media for only 7 days.

The Kaltura Media Access Control Model

Kaltura
open source video

Media Access Control v1 and v2

- Before accessControlProfile there was accessControl
- As of the Gemini version, the KMC application is using accessControl to create and apply Access Control profiles
- Both v2 and v1 can be used, avoid confusion by applying only one of the versions per account

accessControl API (deprecated)	accessControlProfile API (new)
Used by the Kaltura Management Console (KMC) application	Not currently in use by Kaltura applications
Enables restrictions for: domain, ip ranges, countries or sview privileged KS	In addition to v1 restrictions, provides restrictions for: user auth status, metadata logic and user agents
Applied on an entry	Applied on action level (Download, Play and Thumbnail)

The Kaltura Media Access Control Model

Kaltura
open source video

Entitlements

- End-user and category-level “Access Control”
- Provides a framework for complex authentications e.g.
  - Allow ONLY users from department X to access category X
- Available through API and built in to MediaSpace

Kaltura’s Entitlement Infrastructure Information Guide

Kaltura
open source video

Kaltura Session Privileges

- Session privileges allow apps to limit the user to perform only specific actions.
- Privileges are defined using a comma-separated list of key:value pairs.
- Each key-value pair is a specific privilege:
  - The key is the name of the privilege.
  - The value is the object ID to which the privilege applies.
- The key-value pair format is: key followed by the value, separated by a colon: key:value
- Multiple key-value pairs are separated by commas with no spaces:
  - key:1_value,key:0_value
- Some privileges support a wildcard (*) value (for example, sview:*). A wildcard permits the action for all objects.

[Kaltura's API Authentication and Security - KS Privileges]

Kaltura
open source video

Kaltura Session Privileges - Examples

- Allow access to a specific entry Id (remove limitation is set via Access Control):
  Example: allow access to entry id 0_iuasd7:
  ```
  ks = client.session.start ( userSecret, "myUser",
  KalturaSessionType.USER, partnerID , null, "sview:0_iuasd7");
  ```

- Limit number of action For KS:
  Example: limit number of actions to 4:
  ```
  ks = client.session.start (adminSecret, "myUser",
  KalturaSessionType.ADMIN, partnerID , null, "actionslimit:4");
  ```

- Set Role on the KS:
  Example: set role id 2345 on a ks:
  ```
  ks = client.session.start (adminSecret, "myUser",
  KalturaSessionType.ADMIN, partnerID , null, "setrole:2345");
  ```

- **Format: [privilege1]:[data1],[privilege2]:[data2]**

Note: Setting a Role for USER KS will override the set of permissions that the USER KS has. If you give a USER KS the role of an "administrator" he will have administrator permissions.

http://blog.kaltura.org/create-ks-protected-videos-with-free-preview
http://www.kaltura.org/demos/kaltura-paypal/
http://player.kaltura.com/docs/AccessControlPreview

Kaltura
open source video

Kaltura Session – Consider This

- KS generation can happen both on server and client side
- BUT – don't do it on the end-user client side (JS, Flash)
- USER session vs. ADMIN session (API secret keys)
- The User Credentials and Role (Permissions Set)
- You can manage multiple accounts (Always provide Pid)
- KS expiry (time, end action, actions limit)
- There are 'special accounts' partnerIds that are reserved for internal use:
  - BatchManager = -1
  - Admin Console = -2
  - Hosted Pages = -3
  - Server-wide shared content = 0 (e.g. permission items, flavor params)
  - Default Content = 99 (Duplicated to all new accounts, e.g. categories, entries, conversion profiles, access controls)

Kaltura's API Authentication and Security – The Kaltura Session

Kaltura
open source video

Mobile Applications – KS Consideration

- Never include a KS in your app since it will expire.
- Never include a secret (neither user nor admin) in your app - this will be a security risk
- How do you plan to authenticate your users?
  - If users don't identify themselves (public facing, no restrictions, anonymous access) –
    - Create a non-admin user in Kaltura, assign a role only with the permissions needed (e.g. Allow: list, view, upload. Deny: delete, analytics)
    - Compile the user name and password of the user into the app.
  - If the app will authenticate against your server –
    - Your server should generate the appropriate KS once the user authenticates.
  - If users are to be managed in Kaltura –
    - The app/website should register (create) non-admin users using the User API
    - The a login screen will authenticate using user/password to User.loginByLoginId over HTTPS

[Kaltura's API Authentication and Security – The Kaltura Session]

Kaltura
open source video

The KS String Components

- **Publisher ID** – A unique identifier allocated to every Kaltura account.
- **User ID** – The id of the user within the publisher account performing the API call.
- **KS Type** (admin / user)
- **Expiry time** – An unix-time integer representing when the KS will expire.
- **Privileges** – The special set of privileges specified for that KS.
- **Random number** – A time based random number to make every KS string unique.
- **ksdata** - An arbitrary data that will be overlooked by Kaltura and can be used to pass additional information on the KS for custom application use.
- **Signature** – The KS is cryptographically signed (one-way SHA-1 hashing algorithm), by hashing all the above with a secret key shared between Kaltura and the publisher.
- KS v2 is also AES encrypted.

Kaltura's API Authentication and Security – The Kaltura Session

Kaltura
open source video

KS Validation and Audit

- The Kaltura (API) Server will validate the KS:
  - Check the signature against the secret of the specific publisher account.
  - Check whether the time has elapsed or the action limit has been reached.
  - Check whether the KS was not explicitly revoked (using session.end).

- Once all the of validations pass:
  - It will only be allowed to call API actions that the specified user_id is authorized to call (Determined by KS Type, by Role, Privileges).
  - It will only be allowed access to objects that the specified user_id is authorized to access (Determined by Entitlements or Access Control).
  - It will only be allowed to access the publisher account specified (or cross-accounts if VAR or Admin Console).
  - The server will set the owning user for the API actions, e.g. any uploaded content will have the user specified in the KS as its owner.

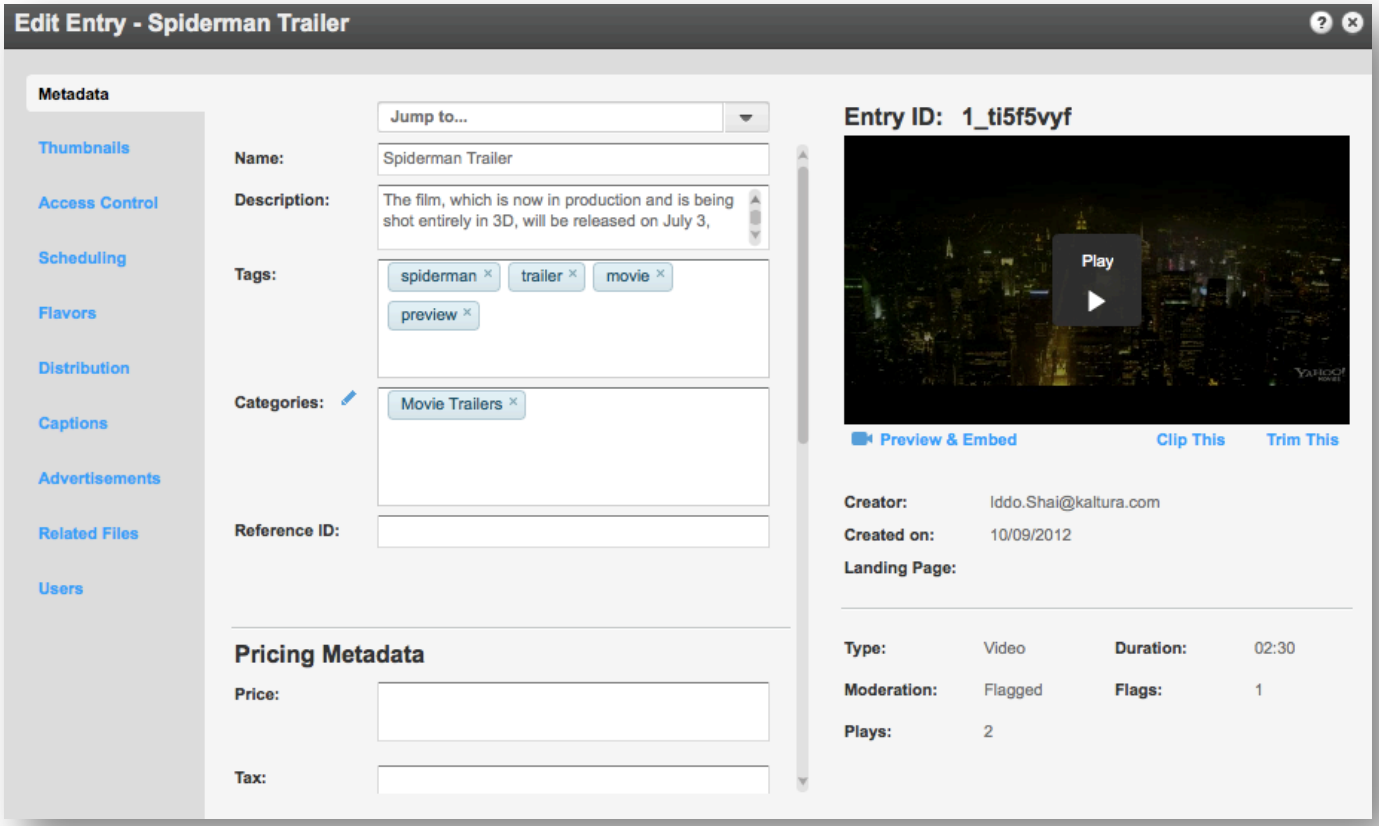
[Kaltura's API Authentication and Security – The Kaltura Session]

Kaltura
open source video

Kaltura Entry Management

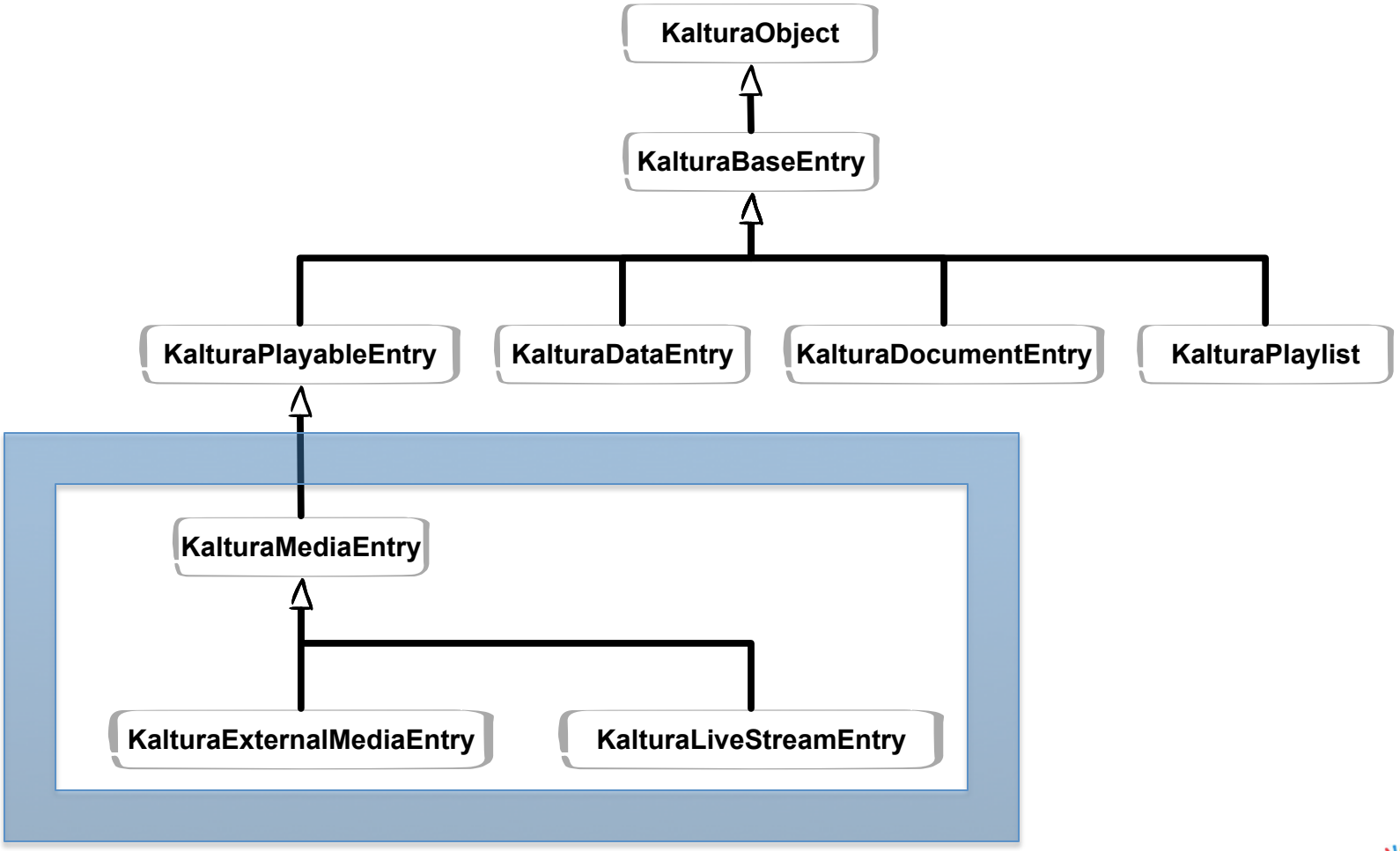
The Kaltura Entry

- Merely a logical wrapper for all aspects of a single media asset (metadata, thumbnails, flavors, etc.)



Kaltura
open source video

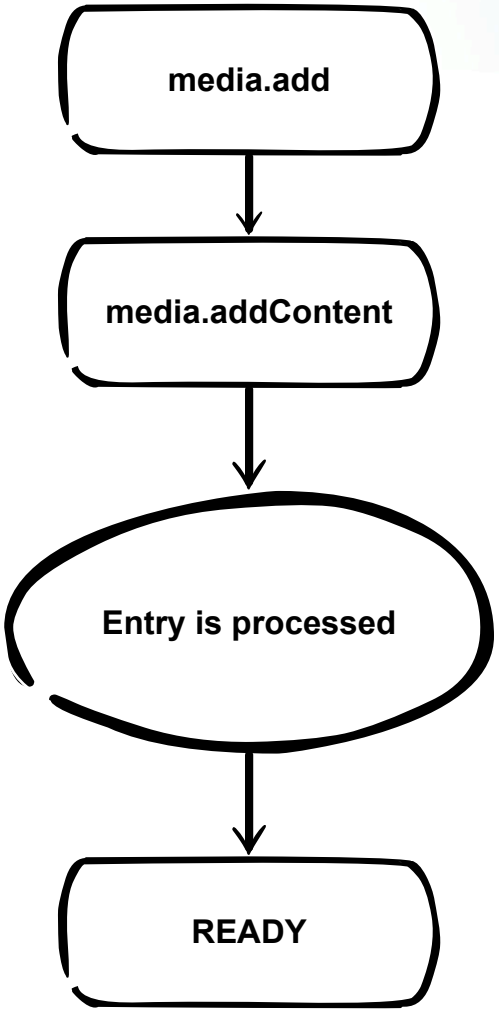
Kaltura*Entry



Kaltura
open source video

Media Entry Life Cycle

- An Entry starts with no media. It's simply an empty metadata container.
- After attaching media to the entry – a whole set of internal processes kicks in
- The Transcoding Profiles assigned to the entry determines when the entry is marked as READY for publishing



Kaltura
open source video

Uploading/Ingesting Content

- Kaltura handles content ingestion from various sources:
  - Upload a file, support pause/resume
  - Import from a URL, support HTTP/s, FTP
  - Import from an SSH URL with keys
  - Import from Remote Storage
  - Import from Drop Folders, support Aspera FASP
  - Webcam recording (Import from AMS)
  - Clone another Kaltura Entry

Kaltura
open source video

Adding a new Entry and Uploading Media

Handshake to create a Kaltura Session:

```
KalturaConfiguration config = new KalturaConfiguration(PARTNER_ID);
config.ServiceUrl = SERVICE_URL;
KalturaClient client = new KalturaClient(config);
client.KS = client.GenerateSession(ADMIN_SECRET, USER_ID, KalturaSessionType.ADMIN,
PARTNER_ID, 86400, "");
```

Create a new Media Entry to which we'll attach the uploaded file:

```
KalturaMediaEntry mediaEntry = new KalturaMediaEntry();
mediaEntry.Name = "Media Entry Using C#";
mediaEntry.MediaType = KalturaMediaType.VIDEO;
mediaEntry = client.MediaService.Add(mediaEntry);
```

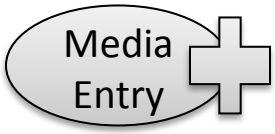
Can also set conversion profile id here

Upload the media File:

```
FileStream fileStream = new FileStream("DemoVideo.flv", FileMode.Open,
FileAccess.Read);
KalturaUploadToken uploadToken = client.UploadTokenService.Add();
client.UploadTokenService.Upload(uploadToken.Id, fileStream);
```

Attach the Media Entry to the File:

```
KalturaUploadedFileTokenResource mediaResource = new
KalturaUploadedFileTokenResource();
mediaResource.Token = uploadToken.Id;
mediaEntry = client.MediaService.AddContent(mediaEntry.Id, mediaResource);
```



[Create A New Kaltura Entry And Upload Video File Using The Kaltura API]

Kaltura
open source video

Updating Entry Details

- media.update – any submitted field will override existing value (no need to media.get before update)

```
String myEntryId = "0_3242das2";
KalturaMediaEntry mediaEntry = new KalturaMediaEntry();
mediaEntry.Tags = "dog,animal,play";
mediaEntry = client.MediaService.update(myEntryId, mediaEntry);
```

Kaltura
open source video

Listing/Searching Entries

- Combine media.list with KalturaMediaEntryFilter
- When using multiple filters the condition will be AND
- Use freeText to search with OR condition across:
  - name, tags, description, id, reference id, and user id
- To create a NOT condition place '!' before the word to exclude. NOT must always follow a positive search first:
  - food that is not pizza == food !pizza
- Un filtered paging is limited to the 10,000 entry. Use a date filter to loop through the entire archive. Example
- Use the advancedSearch property to search inside custom metadata (more on that in Custom Metadata section)

Kaltura
open source video

Categories Management

Kaltura Categories

- A way to organize the library into groups of entries
- An Entry can be associated with 32 Categories (Constant set in entry::MAX_CATEGORIES_PER_ENTRY)
- Categories are organized as a tree
  - Every category have a single parent category
- Entitlements are governed by the Category API -
  - **category** – The Category Service
  - **categoryEntry** – Associate an Entry with Category
  - **categoryUser** – Membership of User in Category

Kaltura
open source video

The Category Service

- Avoid *entry.categories* field (though it's backward compatible)
- To manipulate categories, use the Category service's actions: category.add (KalturaCategory)
- Category has an owner user and moderation settings
- To retrieve a category's full path (to parentId=0), use the fullName and fullIds fields.
  - RootCategory>SecondCategory>ThirdCategory
  - 1765>87623>87612

/index.php?service=category

Kaltura
open source video

Associating Entry with Category

- Use categoryEntry.add (KalturaCategoryEntry)
- Use activate or reject if Moderation is enabled
- To search for all entries in a given category use baseEntry.list combined with:
  - KalturaBaseEntryFilter.categoriesIdsMatchOr
- To search for all categories an entry is associated with use categoryEntry.list combined with:
  - KalturaCategoryEntryFilter.entryIdEqual
- Call the search:
  - categoryEntry.list (KalturaCategoryEntryFilter)

Kaltura
open source video

Associating Users with Category

- Use categoryUser.add (KalturaCategoryUser)
- Use activate or reject if Moderation is enabled
- To search for all members of a given category use categoryUser.list combined with:
  - KalturaCategoryUserFilter.categoryIdIn
- To search for all categories a user is associated with use categoryUser.list combined with:
  - KalturaCategoryUserFilter.userIdIn
- Call the search:
  - categoryUser.list (KalturaCategoryUserFilter)

Kaltura
open source video

Custom Metadata

Kaltura Custom [Meta]Data

- Used to overload new fields on: Entries, Categories, Accounts, Users and Cue Points
- XSD schema Profiles of XMLs
- Every account can have many profiles (xsd schemas) and many fields of various types
- Fields can be set to be indexed, and thus searchable
- In KMC: Closed sets fields will show in the KMC as filters
- In KMC: KMC edit UI can be customized per profile

[How to Add a Kaltura Custom MetaData Schema (Profile)]
[How To Add Fields To Custom Metadata Profile]
[What Fields Can Be Configured In Kaltura Custom Metadata Profiles]

Kaltura
open source video

Kaltura Custom [Meta]Data API

- metadataProfile: manages schema profiles per account
- metadata: manages metadata xmls per object per profile
- The API provides full CRUD management
- Powerful Search API: list (combined with filters)

* For Temporal Metadata – See Cue-Points API

[Creating Profile XSD And Custom KMC View XML For Kaltura Custom Metadata Profile]

Kaltura
open source video

Metadata Schema/Profile - Create

- `metadataProfile.add(metadataProfile, xsdData, viewsData)`
- `xsdData` is the XSD Schema file defining the metadata fields to create
- `viewsData` is a custom definition of KMC UI elements for the newly created fields (if left empty, will use default)
- The Profile is then parsed, indexed and used to build the new fields on the selected object

Creating Profile XSD And Custom KMC View XML For Kaltura Custom Metadata Profile and Sample code (PayPal Gallery Reference App)

Kaltura
open source video

Metadata XML – Create/Update

- `Metadata.add(profileId, KalturaMetadataObjectType, objectId, xml)`
- `profileId` – to which metadata profile this relates?
- `KalturaMetadataObjectType` – what Object?
- `objectId` – the id of the object to add values to
- `xml` – the actual metadata XML (xml must abide the schema/profile XSD and fields order is crucial)
- The xml is then parsed and indexed on the specified object, and is now ready for search/update

Sample code (PayPal Gallery Reference App)

Kaltura
open source video

Metadata XML – Important Notes

- The Metadata schema is **case sensitive.**
  - <price></price> and <Price></Price> are different
- **Order matters**. Make sure to have Metadata XML fields submitted in the same order they appear in the Schema XSD.
- Searchable Date and Integer fields are limited to 4 fields per account (non-searchable is not limited).
- For languages that don't have instance generation from XSD, see the next slide…

Kaltura
open source video

Generate valid metadata XML by the XSD

```
//Get the schema XSD:
$metadataPlugin = KalturaMetadataClientPlugin::get($client);
$schemaUrl = $metadataPlugin->metadataProfile->serve($metadataProfileId); //returns a URL
//or can also use: $metadataPlugin->metadataProfile->get($metadataProfileId)->xsd
$schemaXSDFile = file_get_contents($schemaUrl); //download the XSD file from Kaltura

//Build a <metadata> template:
$schema = new DOMDocument();
$schema->loadXML($schemaXSDFile); //load and parse the XSD as an XML
$fieldsList = $schema->getElementsByTagName('element'); //get all elements of the XSD
$metadataTemplate = '<metadata>'; //Kaltura metadata XML is always wrapped in <metadata>
foreach ($fieldsList as $element) {
    if ($element->hasAttribute('type') == false) continue; //valid fields will always have type
    $key = $element->getAttribute('name'); //systemName is the element's name, not key nor id
    $metadataTemplate .= '<' . $key . '>' . '</' . $key . '>';
}
$metadataTemplate .= '</metadata>';

//Manipulate the template adding your fields values:
$metadataXml = simplexml_load_string($metadataTemplate);
$metadataXml->Price = 2.3; //field's systemName = value;
```

Kaltura
open source video

Submit the Metadata XML

```
$objectType = KalturaMetadataObjectType::ENTRY; //what object will we set metadata for?
$objectId = '0_3242sdf'; //the id of the object you wish to update
$metadataPlugin->metadata->add($mProfileId, $objectType, $objectId, $metadataXml->asXML());
```

```
<metadata>
        <field1>value</field1>
        <field2>value</field2>
</metadata>
```

Kaltura
open source video

Metadata XML – Retrieve

- Metadata XML is attached to the object per profile
- To retrieve the metadata of specific object, use list with filter of, object type, object Id and profile Id

```
filter = KalturaMetadataFilter()
filter.objectIdEqual = objectId
filter.metadataProfileIdEqual = profileId
filter.metadataObjectTypeEqual = objectType //KalturaMetadataObjectType
results = client.metadata.list(filter).objects
```

- This will return a single result if that object has metadata values or an empty result if no metadata XML exist

Sample code (PayPal Gallery Reference App)

Kaltura
open source video

Object Search By Metadata Fields Values

- Search objects using the ***list*** action
- In the filter parameter, set advancedSearch to KalturaMetadataSearchItem, select search type (AND/OR) and set the metadata Profile Id to search on
- Add KalturaSearchCondition or KalturaSearchComparableCondition objects to define search criteria
- Search items and conditions can be chained and nested to create complex search filters
- Use metadataProfile.listFields to get all searchable xPath fields

[How To Search Kaltura Entries By Values Of Custom Metadata Fields Using The API]

Kaltura
open source video

Object Search By Metadata Fields Values

- For more complex queries where search is across multiple metadata profiles, set the advacedSearch attribute to KalturaSearchOperator and chain as many KalturaMetadataSearchItem as needed.

[How To Search Kaltura Entries By Values Of Custom Metadata Fields Using The API]

Kaltura
open source video

Bulk Upload & Content Migration

Bulk Ingestion of Content in to Kaltura

- Bulk Upload API provides a method to perform bulk operations on entries and users.
- There are two methods: CSV and XML. XML is more robust and is the recommended bulk upload method.
- The features supported by the XML method are described in the XML Bulk Upload XSD.
- Use media.bulkUploadAdd action to submit a Bulk Upload XML

Bulk Upload Using CSV or XML | Kaltura Video Tutorial
Kaltura's Bulk Upload XML Content Migration Sample Scripts

Kaltura
open source video

Full Library Export

- ALL objects in Kaltura can be retrieved via the API including entries, categories, users, flavors, profiles, etc.
- Using the API, it is possible to 'dump' a complete Kaltura account for analysis or archival purposes.
- **Note!** When calling the list API action using a pager, there is a hard limit that prevents reaching beyond the 10k object. Instead, to traverse the complete Kaltura account, use a filter with objects sorted by dateCreated and pager of 500 objects in size.

An export utility for exporting full library metadata from Kaltura accounts to Excel file

Kaltura
open source video

Archiving / Distribution Connectors

- When an archival system or 3rd party destination site such as YouTube is used, Distribution Connectors provide a strong mechanism to automatically keep the 3rd party system synchronized with Kaltura.
- Distribution Connectors are Kaltura Server Plugins
  - Enabled via the Admin Console per account
  - Are triggered via API (or KMC), can be set to automatic sync or manual per-entry sync

Kaltura Distribution Module
Creating a Custom Distribution Destination Plugin - Developer Guide

Kaltura
open source video

The Multirequest API

Reducing API Requests

- Often applications have to perform the same series of API requests – Multirequest provides the mechanism to turn this series into a single request, reducing API round-trips.
- It is smart enough to chain output of actions to input of consecutive actions.
- Like any API – Multirequest calls are cached too.
- Using a client library, it's as simple as adding:
  - client.StartMultiRequest();
  - Your code here
  - KalturaMultiResponse response = client.DoMultiRequest();

Kaltura API Usage Guidelines - The MultiRequest API

Kaltura
open source video

Chaining Output to Input

- Use curly-braces to indicate binding
  - {num:result:porpertyName}
  - num - The number of the output request
  - :result: - Will be replaced with the result of [num]
  - properyName - The value of *property* from [num]
- See examples in the client libraries sample codes

Kaltura API Usage Guidelines - The MultiRequest API

Kaltura
open source video

Transparent Multirequest Mapping

- The PHP and Python Client Libraries can automatically generate the chaining string based on objects and fields.

```
//The following example -
$thumbUrl = "http://mysite.com/thumbnail.jpg";
$client->startMultiRequest();
$mediaEntry = new KalturaMediaEntry();
$mediaEntry->name = "Sample media";
$mediaEntry->mediaType = KalturaMediaType::VIDEO;
//1st call: add an entry
$mediaEntry = $client->baseEntry->add($mediaEntry); //important that you'll assign result
//2nd call: update the entry thumbnail:
$client->baseEntry->updateThumbnailFromUrl("{1:result:id}", $thumbUrl);
$client->doMultiRequest();
//the 2nd call can also be written as: (transparent mapping)
$client->baseEntry->updateThumbnailFromUrl($mediaEntry->id, $thumbUrl);
```

Kaltura
open source video

The PlayManifest & Download URL

The PlayManifest API

- The playManifest action purpose is to generate stream URLs for media player applications.
- playManifest features several return types, mainly:
  - http redirect to the video file F4M manifest
  - XML used by the flash OSMF player (HTTP / RTMP)
  - SMIL manifest - XML used when playing with Akamai HD in flash
  - M3U8 manifest - master M3U8 for Apple HTTP live streaming (HLS)

How to retrieve the download or streaming URL using API calls?

Kaltura
open source video

Dynamic Thumbnails & thumbAsset API

Dynamic Thumbnails

- The Thumbnail API provides a simple web interface to dynamically generate image snapshots of Kaltura video entries on the fly.
- The images are generated upon demand (with caching on disk and via CDN) by calling the following URL -
- http://YourKalturaServer/p/{partner_id}/thumbnail/entry_id/{entry_id}

The Kaltura Thumbnail API

Kaltura
open source video

Editorial Thumbnail Assets

- Editorial thumbnail assets are static images stored in Kaltura per video entry.
- Video entries can have an unlimited number of associated editorial thumbnails.
- To manage thumbnail assets, use the thumbAsset API.
- Thumbnail Assets can be:
  - An uploaded image
  - A captured manipulated image from the source video (using thumbParams)

The Kaltura Thumbnail Assets API

Kaltura
open source video

Server Notifications

Kaltura Server Side Notifications

- HTTP POST requests sent by the Kaltura server
- Notify client server of any Kaltura entry changes
- Can be distributed using a script on the endpoint
- /api_v3/testmeDoc/index.php?page=notifications
- Use the notifications client for easier handling (PHP/Java)

How To Handle Kaltura Server Notifications in PHP

Kaltura
open source video

The Events

Event	Description
Entry Add	An entry was successfully added the account.
Entry Block	An entry was blocked by a moderator or admin user.
Entry Delete	An entry was deleted.
Entry Update	An entry was updated/modified.
Entry Update Moderation	An entry's moderation status was changed.
Entry Update Thumbnail	The default thumbnail of the entry was replaced.
Entry Update Permissions	An entry's privacy settings were changed.
User Add	A new user was added to the Kaltura account.
User Ban	A user was banned from the Kaltura account.

/api_v3/testmeDoc/index.php?page=notifications

Kaltura
open source video

KMC – Notification Setup

Index of / | www.kaltura.com/api_v3/ | Kaltura - TestMe Console | KMC > account > integra | Kaltura - TestMe Console

www.kaltura.com/index.php/kmc/kmc4#account|integration

Kaltura Admin Cons... | Expense Report Aut... | Kaltura SharePoint | Open Source Trainin... | Open Source - Face... | Building Communit... | Other bookmarks

Kaltura | Dashboard | Content | Studio | Analytics | Settings | Administration | Upload | Zohar Babin

Account Settings | Integration Settings | Access Control | Transcoding Settings | Custom Data | My User Settings | Account Upgrade

Account Info

Partner ID:	190182
Sub Partner ID:	19018200
Administrator Secret:	435c4439888ffa864e955dc81a6b8f49
User Secret:	daf503d2868e8a39a5e4970480a65c4f

Advanced Settings

Notifications

Do you want to receive server notifications? Yes No

Enter Notification URL: http://www.kaltura.org/testnoti/notific Ping URL

Notification Name	Sent by Client	Sent by Server
Add Entry	☐	☑
Update Entry Permissions	☐	☑
Delete Entry	☐	☑
Block Entry	☐	☑
Update Entry	☐	☑
Update Entry Thumbnail	☐	☑
Update Entry Moderation	☐	☑

Save Changes

Kaltura open source video

Example notification in action

```
Array
(
    [notification_id] => 417297831
    [notification_type] => entry_update
    [puser_id] => example
    [partner_id] => 715232
    [entry_id] => 0_14bvzhzx
    [name] => Home
    [tags] => homepage, creative
    [search_text] => Home homepage, creative Homepage video with music + notification test
    [media_type] => 1
    [length_in_msecs] => 52297
    [permissions] =>
    [thumbnail_url] => http://cdnbakmi.kaltura.com/p/715232/sp/71523200/thumbnail/entry_id/0_14bvzhzx/version/100000
    [kshow_id] =>
    [group_id] =>
    [partner_data] =>
    [status] => 2
    [width] => 528
    [height] => 352
    [data_url] => http://cdnbakmi.kaltura.com/p/715232/sp/71523200/flvclipper/entry_id/0_14bvzhzx/version/0
    [download_url] => http://cdnbakmi.kaltura.com/p/715232/sp/71523200/raw/entry_id/0_14bvzhzx/version/0
    [download_size] => 1697591
    [media_date] =>
    [moderation_status] => 6
    [ks_data] =>
    [signed_fields] => notification_id,notification_type,puser_id,partner_id,entry_id,name,tags,search_text,media_type,
                       length_in_msecs,permissions,thumbnail_url,kshow_id,group_id,partner_data,status,width,height,
                       data_url,download_url,download_size,media_date,moderation_status,ks_data,signed_fields,
    [sig] => 5b06d8b06bb40fe2bc31718ef1bc7f2b
)
```

Kaltura
open source video

Kaltura Players And Player API

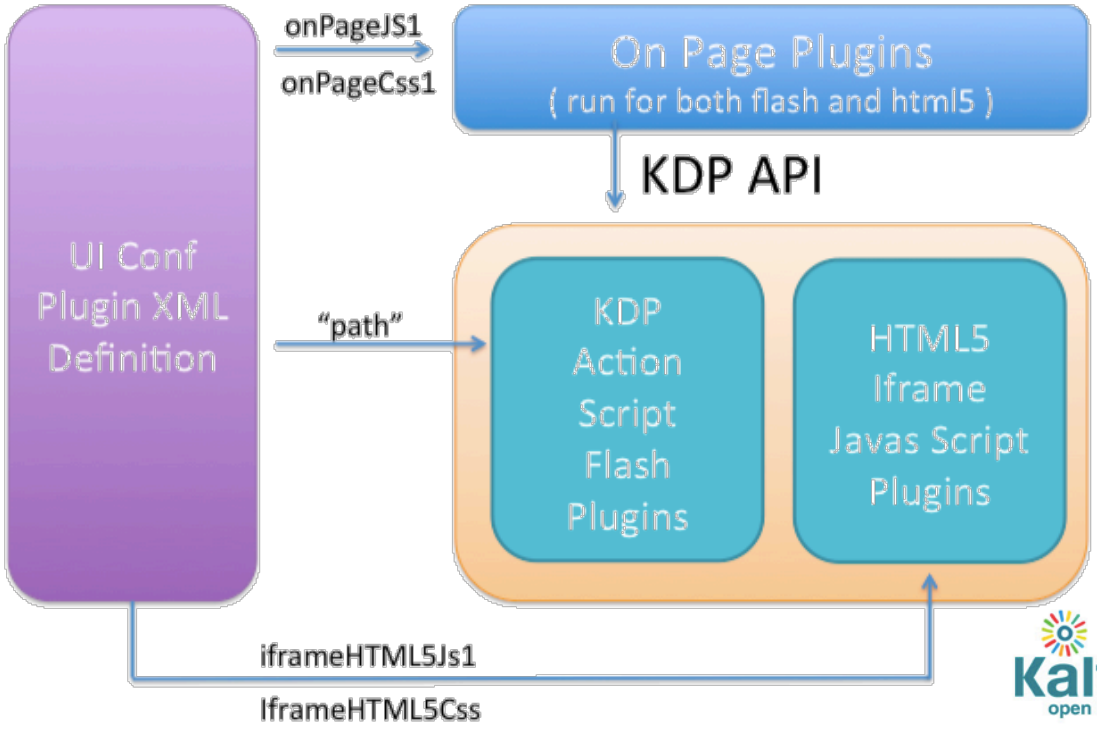
Kaltura Player

Demos, Configuration Options and Tools for Creating Kaltura Player Experiences

Kaltura
open source video

Kaltura Player Framework

- Open Source and Widely Used.
- **Unified Flash and HTML5 API**
- Intelligent per-platform player selection

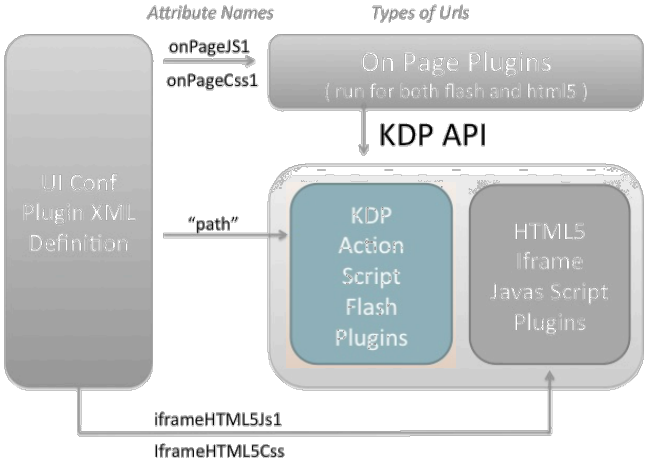


http://html5video.org

Kaltura
open source video

Kaltura Flash Player – aka KDP3

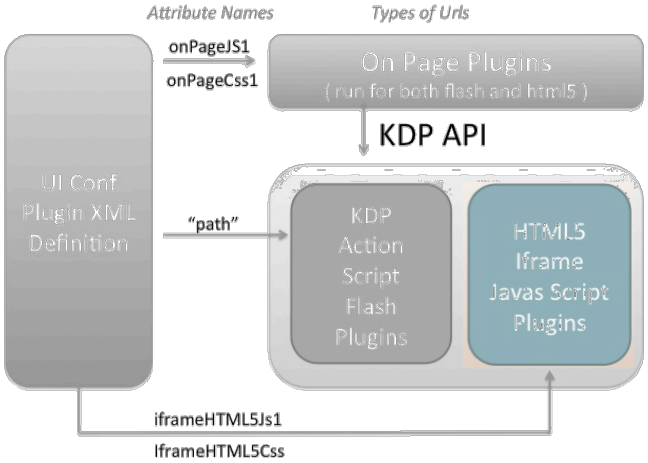
- Kaltura Dynamic Player is a Flash based media player
- An XML configuration file defines its layout, behavior and visual elements/skin. The XML is dubbed uiConf
- Flash KDP is based Open Source Standards:
  - Adobe OSMF
  - PureMVC
  - Yahoo! Astra
  - Fl Components
  - Kaltura ActionScript 3 Client Library



[Introduction to Kaltura's Cross-Platform Media Players]

Kaltura HTML5 Player

- HTML5 (JavaScript, CSS and HTML) based media player
- Intelligent per-platform video flavor and feature selection
- Based on Open Source standards:
  - jQuery
  - MediaWiki Resource Loader
  - CSS Based Skins



Introduction to Kaltura's Cross-Platform Media Players

Kaltura
open source video

Setup and Embed

- AutoEmbed – Single line embedding:

```
<script src="http://[server-url]/p/[partner_id]/sp/[partner_id]00/
embedIframeJs/uiconf_id/[uiconf_id]/partner_id/[partner_id]?
autoembed=true&entry_id=[entry_id]&playerId=kaltura_player&width=40
0&height=330"></script>
```

- Robust Embedding Options:
  - AutoEmbed: **Single line of script!** Great for quickly getting a player on the page.
  - Dynamic Embed: Great for code integration in web apps. Responsive web-design friendly. Cleaner configuration.
  - Thumbnail embed: Fast loading, great for pages with lots of players. Best used in video-blogs.
  - Iframe: **Single line of HTML!** Great for when JS is not allowed. Best option for Share/Embed players.

Kaltura Embed Code Generator

Kaltura
open source video

Player Embed API

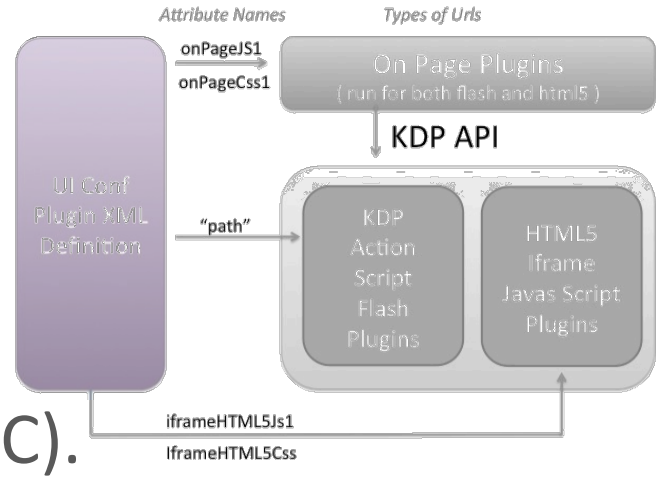
- kWidget.embed – Dynamic Embed
- kWidget.thumbEmbed – Thumbnail Embed
- kWidget.getKalturaThumbUrl – Get thumbnail URL, settings is an object with Kaltura Thumbnail attributes
- kWidget.addReadyCallback – Add event listener, listen to when players on the page are ready for JS binding
- kWidget.destory – Removes the player from the DOM

https://github.com/kaltura/mwEmbed/blob/master/kWidget/kWidget.js

Kaltura
open source video

Library Configurations

- Configurations are stored on the server as uiConf XML.
- 3 ways to modify configurations:
  - Modify the uiConf XML file.
  - uiVars configurations section:
    - Edit in the uiConf XML.
    - Edit in the Player Studio (KMC).
  - Edit the flashVars on the page.



- They all affect the same uiConf XML.
- Order of significance:
  - uiConf > uiVars > flashVars > uiVars override

[How to use the Kaltura Player Studio's Additional Parameters and Plugins (UIVars, flashvars)]

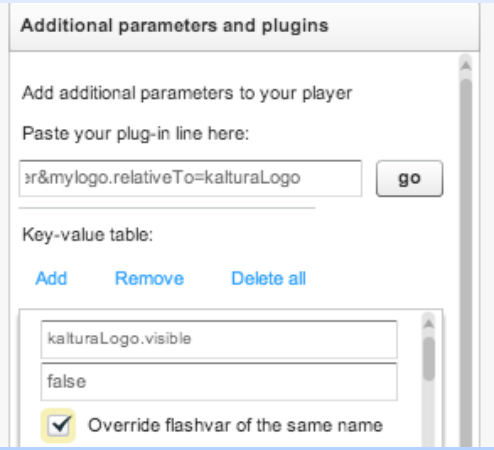
Kaltura
open source video

Configuration Example: Logo

uiConf XML

```
<Plugin id="mylogo" width="30" height="30"
        includeInLayout="false" relativeTo="kalturaLogo"  position="after"
        className="Watermark"
        watermarkPath="http://myserver.com/mylogo.png"
        watermarkClickPath="http://www.myserver.com/"      />
```

uiVars Player Studio



flashVars

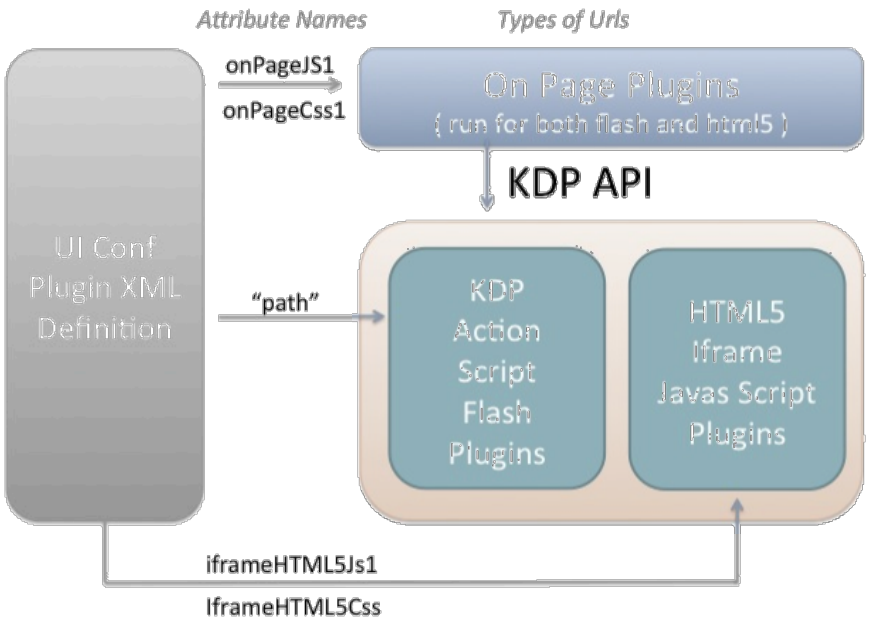
```
flashvars: {
    "mylogo": {
            "plugin" : true,
            "width" : "30",
            "height" : "30",
            "includeInLayout" : false,
            "relativeTo" : "kalturaLogo",
            "position" : "after",
            "className" : "Watermark",
            "watermarkPath" : "http://myserver.com/mylogo.png",
            "watermarkClickPath" : http://www.myserver.com
        },
},
```

[How to use the Kaltura Player Studio's Additional Parameters and Plugins (UIVars, flashvars)]

Kaltura
open source video

Player JavaScript API

- Unified Flash and HTML5 API
- For API docs see: KDP JavaScript API
- For samples see: The Player Features Hub



Kaltura
open source video

Player JavaScript API – KDP

- kBind – listen to player events
- kUnbind – remove a listener
- sendNotification – perform an action
- evaluate – get a value of a player object's attribute
- setKDPAttribute – set a value in player object's attribute

* The player has built in parameters binding mechanism, allowing for evaluation using curly braces

Kaltura
open source video

Player JavaScript API - Basic Example

- Log Player Id to Console, once Player is Ready to Play

```
kWidget.embed({
    'targetId': 'myVideoTarget',
    'wid': '_243342',
    'uiconf_id' : '2877502',
    'entry_id' : '0_uka1msg4',

     // Ready callback is issued for this player:
    'readyCallback': function( playerId ){
       console.log( "Kaltura player is ready for API bindings: " + playerId );
       var kdp = $( '#' + playerId ).get(0); //get the player Id
       kdp.kBind( 'mediaReady', function(){ //bind to the mediaReady notification
          console.log(  "Kaltura player is ready to play! " + playerId );
       });
    }
});
```

Kaltura
open source video

Player JavaScript API - Basic Example

- addReadyCallback will be called for all players on the page:

```
kWidget.addReadyCallback (function( playerId ){
    console.log( "Kaltura player is ready for API bindings: " + playerId );
    var kdp = $( '#' + playerId ).get(0); //get the player Id
    kdp.kBind( 'mediaReady', function(){ //bind to the mediaReady notification
        console.log(  "Kaltura player is ready to play! " + playerId );
});

kWidget.embed({
    'targetId': 'myVideoTarget',
    'wid': '_243342',
    'uiconf_id' : '2877502',
    'entry_id' : '0_uka1msg4'
});
```

Kaltura
open source video

Player JavaScript API - Basic Example

- On Playback Progress, log to console the current play head time:

```
kWidget.addReadyCallback (function( playerId ){
    console.log( "Kaltura player is ready for API bindings: " + playerId );
    var kdp = $( '#' + playerId ).get(0); //get the player Id
    kdp.kBind( 'playerUpdatePlayhead', function(){ //bind to the progress notification
        var currentTime = kdp.evaluate('{video.player.currentTime}');
        console.log(  "Current playhead time: " + currentTime );
});

kWidget.embed({
    'targetId': 'myVideoTarget',
    'wid': '_243342',
    'uiconf_id' : '2877502',
    'entry_id' : '0_uka1msg4'
});
```

Kaltura
open source video

Player API – Common Examples

- Play – sendNotification(‘doPlay’);
- Pause – sendNotification(‘doPause’);
- Seek – sendNotification(‘doSeek’, 20.3); //float seconds
- Get the player’s title and description –
  - evaluate('{mediaProxy.entry.name}');
  - evaluate('{mediaProxy.entry.description}');
- Change the playing entry –
  - sendNotification(‘changeMedia’, {entryId: ‘0_as23dcx’});
- Set volume - sendNotification('changeVolume', 1); //float 0-1
- Listen for end of playback –
  - kBind('playerPlayEnd', function() {
    console.log("Kaltura player finished playback");
    }
- Move to the next item in the playlist

Kaltura
open source video

Calling the Kaltura API in JS - kWidget.api

- doRequest – getting an entry in JavaScript

```
//instantiate a new API object:
var kapi = new kWidget.api( { 'wid': '_243342' });
//Make an API call:
kapi.doRequest(
    {  //the request and its parameters:
        'service': 'baseEntry',
        'action': 'get',
        'entryId': '0_uka1msg4'
    },
    //the response handler:
    function( data ){
        // get the result from the API response
        var entry = data;
        // output the data:
        console.log(entry);
    }
);
```

Kaltura
open source video

Calling the Kaltura API in JS - kWidget.api

- doRequest – get entry's custom metadata in multirequest

```
kapi.doRequest([{
    'service': 'baseEntry',
    'action': 'get',
    'entryId': '0_uka1msg4'
}, {
     'service' : 'metadata_metadata',
     'action' : 'list',
     'version' : '-1',
     // metaDataFilter:
     'filter:metadataObjectTypeEqual' :1, /* KalturaMetadataObjectType::ENTRY */
     'filter:orderBy' : '+createdAt',
     'filter:objectIdEqual' :'0_uka1msg4',
     'pager:pageSize' : 1
}], function( data ){
        var xml = $.parseXML( data[2].objects[0].xml );
        var $xml = $( xml ).find('metadata').children();
        var customMeta = {};
        $.each( $xml, function(inx, node){
            customMeta[ node.nodeName ] = node.textContent;
        });
        var entry = data[1];
        // output the data:
        console.log( customMeta, entry);
});
```

Kaltura
open source video

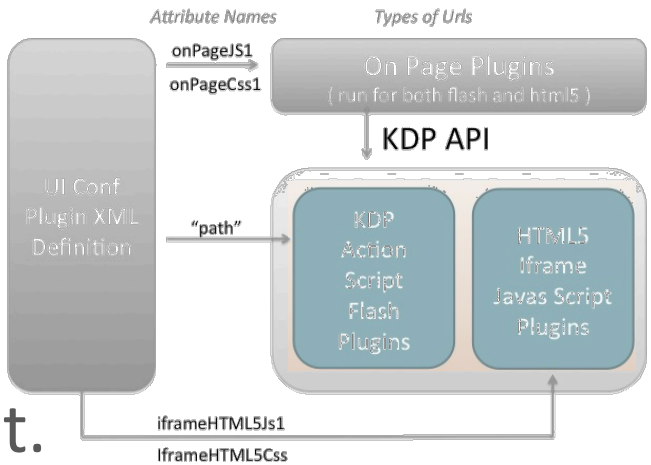
Plugins - Which side of the player?

- In-Player Plugins (aka iFrame plugins in HTML5)
  - JavaScript or Flash plugins loaded inside the Player
  - Have to write two plugins: one for Flash and one for JavaScript
  - Allows greater control and flexibility
- On-Page Plugins
  - JavaScript loaded on the hosting page
  - Leverage the unified kdp and kWidget APIs
  - Write once! Works for both Flash and HTML

Kaltura
open source video

In-Player Plugins

- In-Player plugins are good when access is required to:
  - Full-screen support
  - Manipulate playback sequence
    e.g. Advertising
  - Manipulate streaming
    e.g. peer2peer Streaming, CDN int.

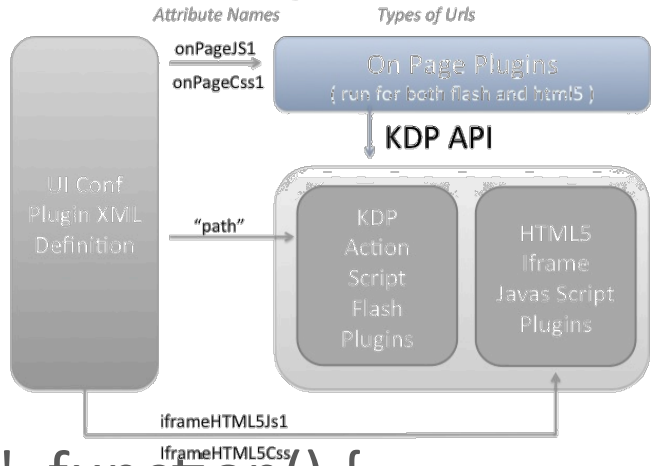


- Must write 2 different plugins:
  - KDP3 – The Flash Player AS3 plugin
  - HTML5 – The iFrame JS plugin

Kaltura
open source video

On-Page Plugins – Getting Started

- On-Page plugins will **NOT** work on full-screen and will **NOT** have access to manipulate playback or streaming



- To create an on-page plugin:
  - (){ //your plugin here })();
  - kdp = $('#' + playerId).get(0);
  - Instantiate after mediaReady:
    kdp.kBind('mediaReady.*myPlugin*', function() {
    new myPlugin(playerId);
    }
  - Namespace your listeners!
  - Use kWidget.log instead of console.log

Boiler-Plate: Kaltura OnPage Plugins
Example: The Video Details Block Plugin

Kaltura
open source video

Loading an On-Page Plugins

- Include the plugin configurations in flashVars as a new plugin object

```
"flashvars":{
        //this is how we load our custom on-page plugin:
        'videoDetailsBlockPlugin':{
            'plugin':true,
            'path':"/content/uiconf/ps/kaltura/kdp/v3.6.9/plugins/facadePlugin.swf",
            'relativeTo':'video',
            'position': 'before',
            'includeInLayout': false,
            'requiresJQuery': true,
            'onPageJs1':'onpage-plugin/onpage-plugin.js',
            'onPageCss1':'onpage-plugin/onpage-plugin.css',
            'width':'0%',
            'height':'0%',
            'customVar':1287924
        },
```

Enable the plugin

Support Flash

Add jQuery support

Load the plugin

Your custom configs

Boiler-Plate: Kaltura OnPage Plugins
Example: The Video Details Block Plugin

Kaltura
open source video

kWidget Utility Functions

- Load external JavaScript - .appendScriptUrls(urlsArray, callback)
- Load external CSS - .appendCssUrl(url)
- Detect IE or iOS - .isIE() & .isIOS()
- Support for HTML5 - .supportsHTML5()
- Support for Flash - .supportsFlash() & .getFlashVersio()
- Write a message to the console.log - .log(msg)

kWidget source code

Kaltura
open source video

Create Your On-Page Plugin

- Fork the mwEmbed library to your own repository
- Follow the library setup guidelines
- Open /mwEmbed/kWidget/onPagePlugins
  - See videoDetailsBlock plugin as reference sample
- Create your plugin in a new folder under onPagePlugins
- Write unit tests in the *.qunit.html file to verify your plugin works well and will not break in future releases
- When you're done – Create a pull-request for review

Boiler-Plate: Kaltura OnPage Plugins
Example: The Video Details Block Plugin

Kaltura
open source video

Thank You!

Kaltura
open source video