Ooyala Video Plugin

A Technical and User Guide

Table of Contents

OVERVIEW

This document describes the technical implementation of the Ooyala video component and Assets search panel in Edit mode on Adobe's AEM 6.2 and Adobe's AEM 6.3. This functionality is packaged following the "Package Manager" tool embedded in Adobe's AEM 6.2 and Adobe's AEM 6.3. The package contains the following items:

1. Ooyala assets search panel in Edit mode
2. Ooyala Video Component
3. Ooyala Configuration Page
4. Ooyala Uploader Page
5. OSGI Bundle with Java classes

The Ooyala plugin enables content authors to find videos and playlists from Backlot and place those on pages. Users can configure the video component with players and player sizes. The V2 API's are used to create the Ooyala plugin and are detailed at the following location: http://api.ooyala.com/docs/v2/

INSTALL

The Ooyala AEM plugin (the plugin) has been developed using AEM's standard component model. The plugin consists of an OSGi bundle/module to handle Ooyala Backlot/REST API integration, the Ooyala Video Component (HTL and Dialog), Tools Configuration Page, Uploader Page and "Ooyala" assets search panel in Edit mode. The plugin is distributed as an AEM Package for easy installation. The source code is also available for download from https://github.com/ooyala/OoyalaAEM.

The file is downloaded in zip. format.

On the Start page (/projects.html/content/projects – AEM 6.2; /aem/start – AEM 6.3) go to Tools -> Deployment -> Packages. To navigate directly to the Package Manager Page the url is http://[replace with the author instance domain]/crx/packmgr/index.jsp
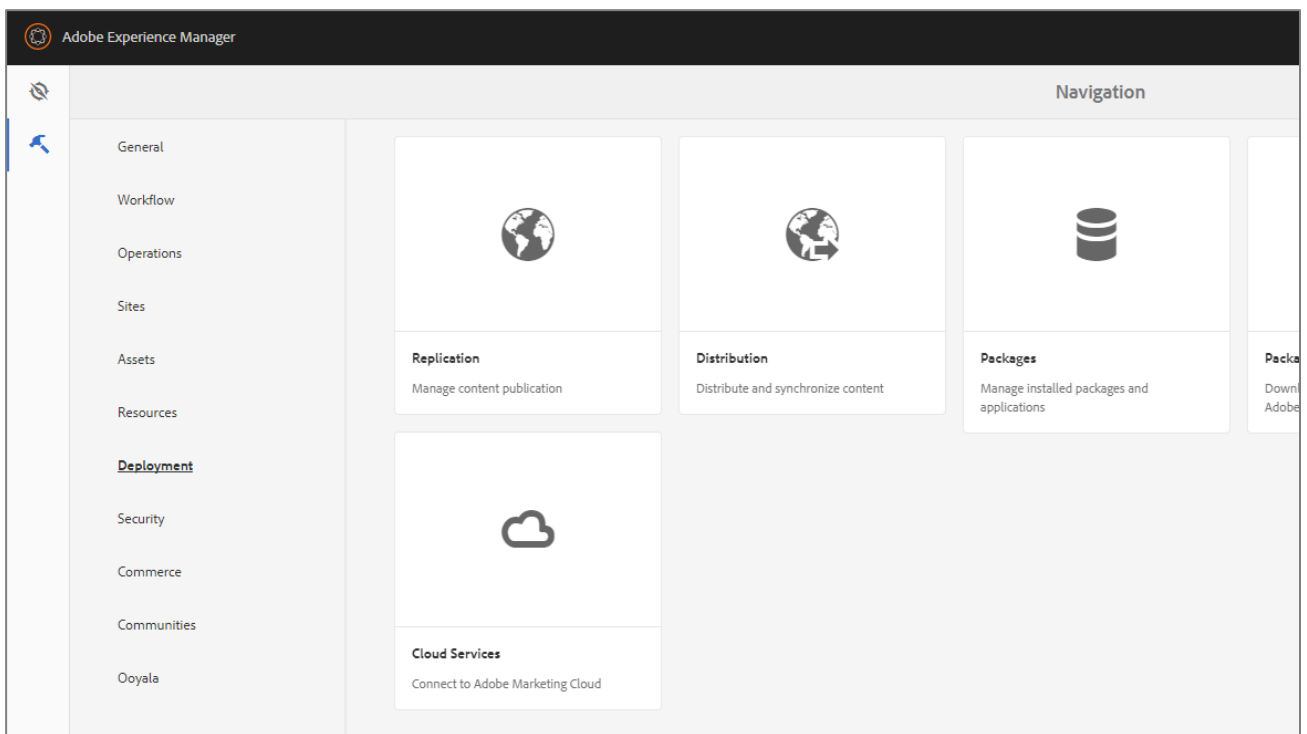
Figure 1

On the package manager screen click "Upload Package" on the top menu bar; this will bring up a wizard to upload the package. Navigate to the folder where the zip file is located. Ensure the zip file is not unzipped. When selected the name of the zip file will be displayed in the Package field. If this is the first time the package is being uploaded, the "Force Upload" can be left blank. In general, it is recommended to click this button to install new versions of the package. Uninstalling the package will delete all the data as well.
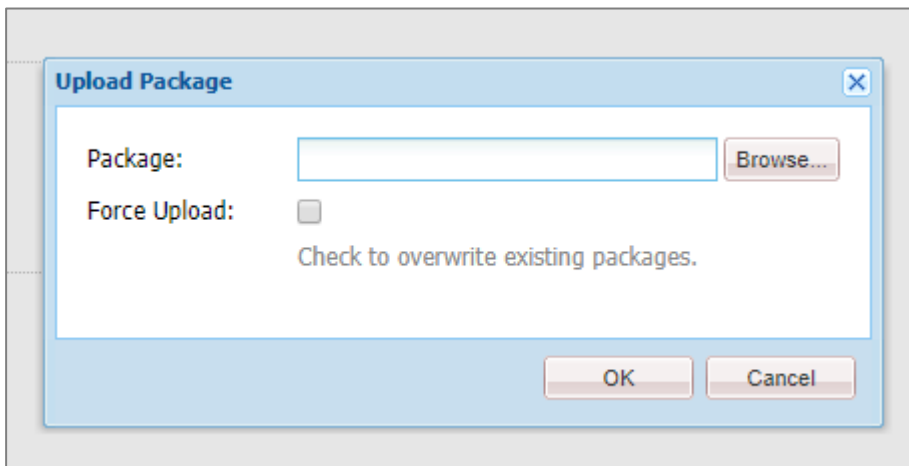
Figure 2

Once the zip file has been uploaded, it needs to be installed. Click on the "install" button on the top right.
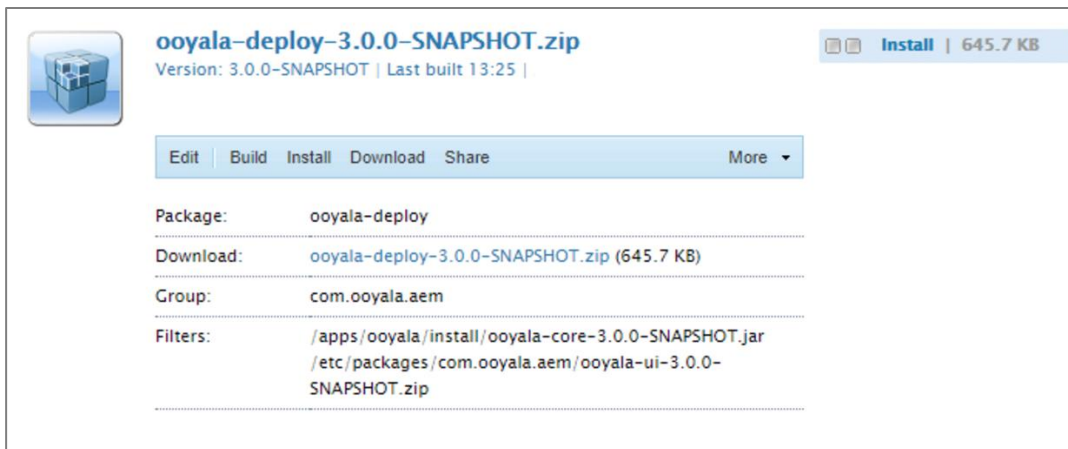

Figure 3

If the install was successful, the following message will be displayed at the bottom of the console.

```
- /etc/ooyala/ooyala-uploader/jcr:content/body/items/form/items/
- /etc/ooyala/ooyala-uploader/jcr:content/body/items/form/items/
- /etc/ooyala/ooyala-uploader/jcr:content/head/clientlibs
- /etc/ooyala/ooyala-uploader/jcr:content/head/favicon
- /etc/ooyala/ooyala-uploader/jcr:content/head/viewport
- /etc/ooyala/synthetic-resource
- /etc/ooyala/synthetic-resource/synthetic-resource
- /etc/map
- /etc/map/ooyala_progress_map
- /home
- /home/users
- /home/users/system
A /home/users/system/ooyala-system-user
- /home/groups
- /home/groups/o
A /home/groups/o/ooyala-system-users
A /home/groups/o/ooyala-system-users/rep:policy
saving approx 6 nodes...
Package imported.

Package installed in 2405ms.
```

Figure 4

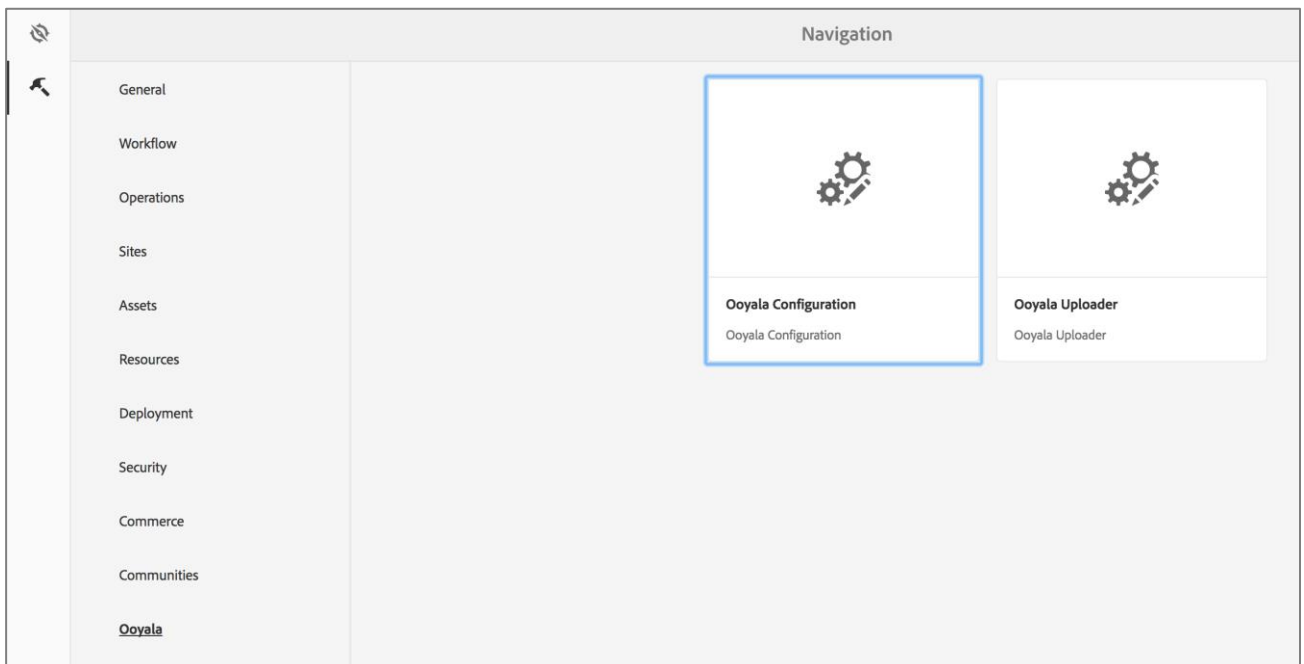Once the package installs successfully go to Tools -> Ooyala.

Figure 5

The following fields are displayed when the dialog loads:
- API Key: the global account key from backlot
- API Secret : the global account key from backlot
- Metadata Keys: The Ooyala API matches metadata keys exactly, so a list of the actual key needs to be configured. This list of keys will be displayed on the "Ooyala" assets search panel in Edit mode, so users can select which metadata field they would like to search on.
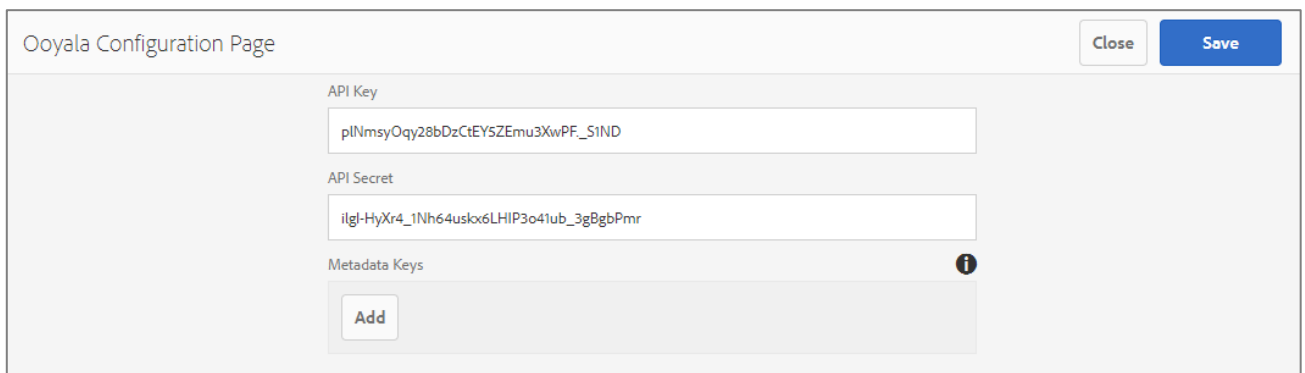
Figure 6

> **NOTE**: *API Key/API Secret are required fields.*

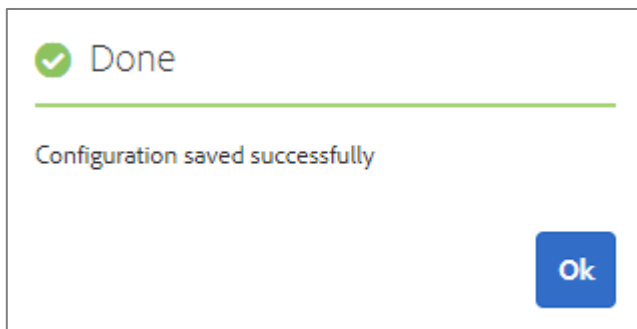Once the configuration is saved, confirmation message will be displayed.

Figure 7

RE-INSTALLING/UPDATING THE PACKAGE

Re-installation of the Ooyala package requires uninstalling the Ooyala OSGi bundle first. Once the OSGi bundle has been uninstalled, the package can be re-installed using the same instructions as the previous (Installation) section.

> **NOTE:** *Re-installing the Ooyala AEM Package will erase the Ooyala Backlot API Credentials on the Ooyala Configuration Page. Make sure to backup your API Key/s, API Secret, and Metadata fields before re-installing the package.*

This procedure requires AEM administrator privileges.

Uninstall the Ooyala OSGi bundle:

- Navigate to the AEM OSGi Web Console: http://[replace with the author instance domain]/system/console/bundles
- Using your browser's search function, find the Ooyala OSGi bundle. Search for "ooyala".
- The bundle should be named Ooyala Adobe AEM Core
- Making sure you have selected the Ooyala OSGi bundle.
- Click on the trash can icon/uninstall.

NOTE: Deleting an incorrect bundle could cause the system to fail.

Re-install the Ooyala AEM Package using the instruction from the previous section.

USING THE COMPONENT

To use the Ooyala Video Component on any page, the page container component needs to allow the "Ooyala Video Component".

Refer to following links to get following information about Editable and Static templates in AEM:

For AEM 6.2:

Editable: https://docs.adobe.com/docs/en/aem/6-2/develop/templates/page-templates-editable.html

Static: https://docs.adobe.com/docs/en/aem/6-2/develop/templates/page-templates-static.html

For AEM 6.3:

Editable: https://docs.adobe.com/docs/en/aem/6-3/develop/templates/page-templates-editable.html

Static: https://docs.adobe.com/docs/en/aem/6-3/develop/templates/page-templates-static.html

To update existing Editable template in AEM 6.2 do the following:

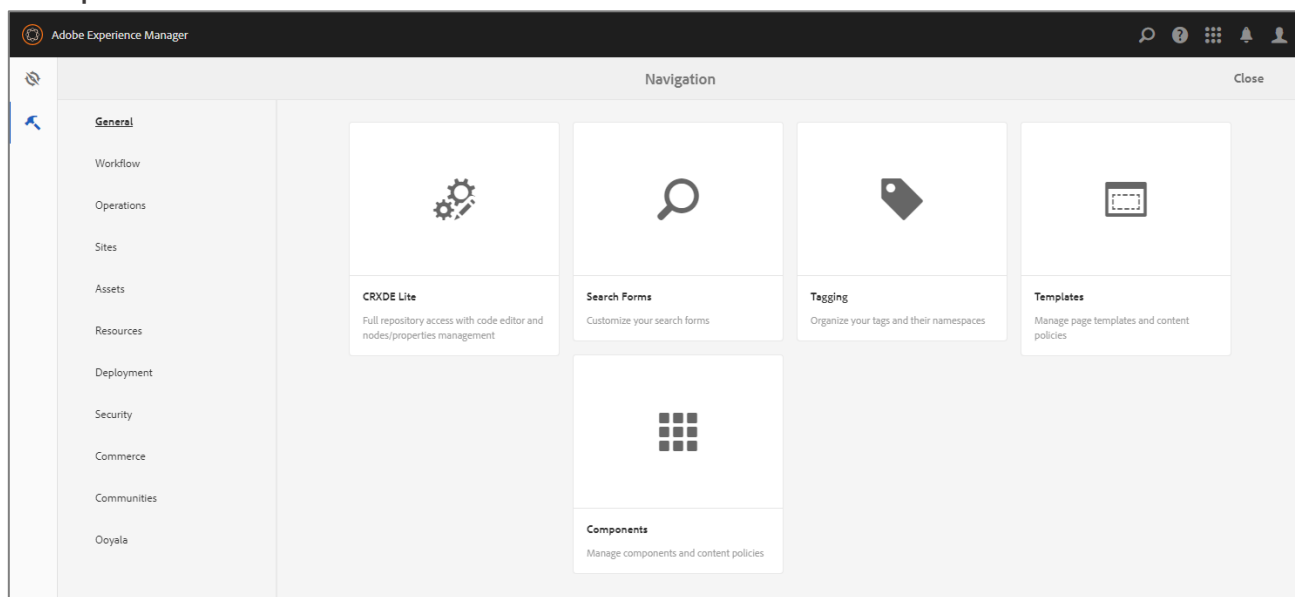1. On the Start page (/projects.html/content/projects) to Tools -> General -> Templates

Figure 8

2. Select template to update and click "Edit" button
3. In "Structure" mode select needed parsys to place component
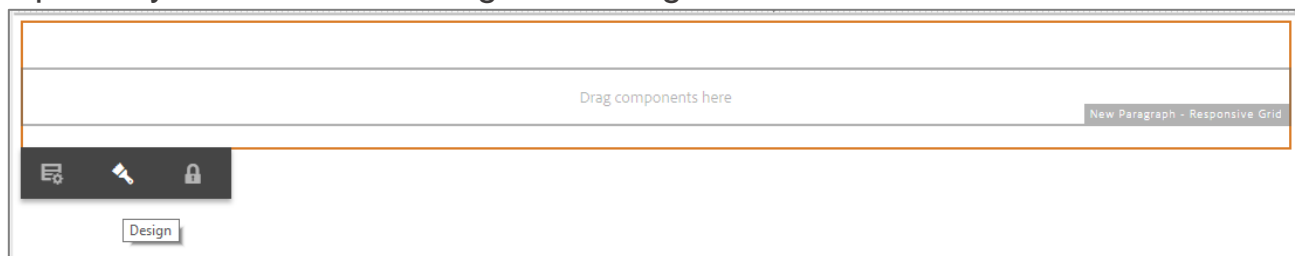4. Open Layout Container Design via Design button

Figure 9

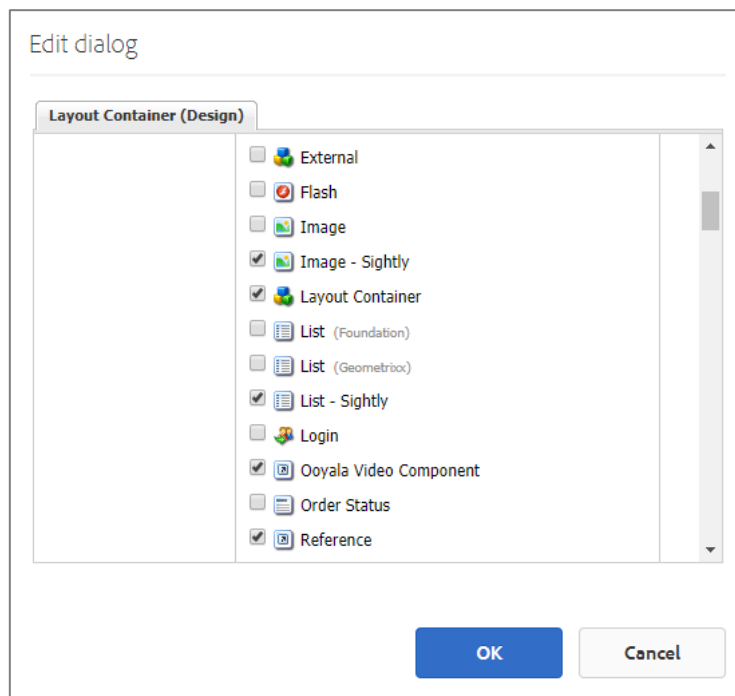5. Expand "General" component group
6. Check "Ooyala Video Component"

Figure 10

7. Save changes pressing "Ok" button
8. Go to any page created by updated template
9. Select "Components" on editor panel
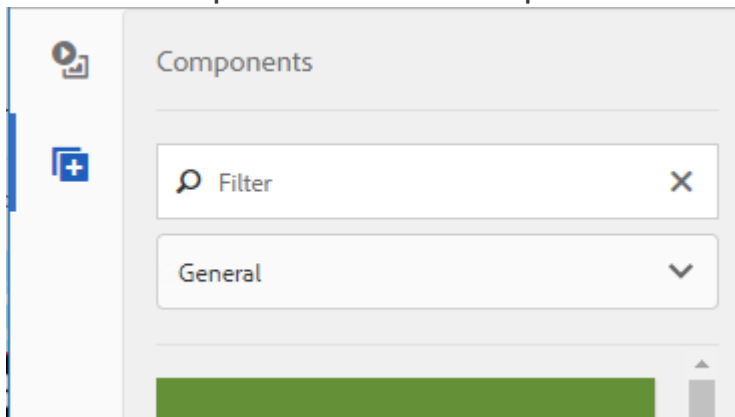
Figure 11

10.   Check "General" component group
11.   Drag and drop "Ooyala Video Component" on the page

To update existing Editable template in AEM 6.3 do the following:

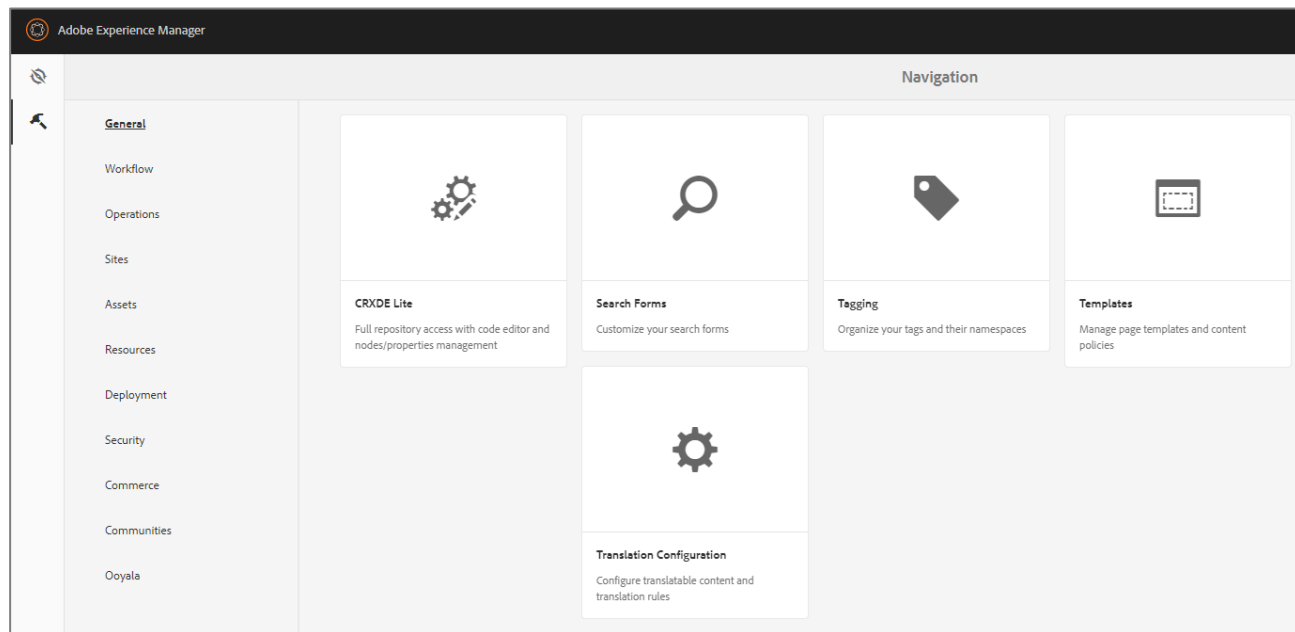1. On the Start page (/aem/start) to Tools -> General -> Templates

Figure 12

2. Select template to update and click "Edit" button
3. In "Structure" mode select needed parsys to place component
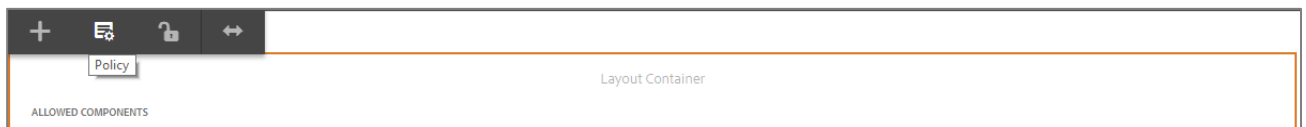4. Open Layout Container Design via Policy button

Figure 13

5. Select Policy to update
6. On the "Properties" panel expand "General" component group
7. Check "Ooyala Video Component"

Properties

Then adjust the settings of the selected policy to configure the component

Allowed Components   Default Components   Responsive Settings

🔍 Search

∨ ▬ **General**

☐ 🖼 Adaptive Image

☐ ≣ AEM Form

☐ Carousel

☐ Chart

☑ Co Content Fragment

☐ Design Importer

☐ Download

☑ Ex Experience Fragment

☐ Ex External

☐ Flash

☐ 🖼 Image

☐ 🖼 Image - HTL

☑ La Layout Container

☐ ≣ List

☐ ≣ List - HTL

☐ Login

☑ Oo Ooyala Video Component

Figure 14

8. Save changes pressing "Done" button
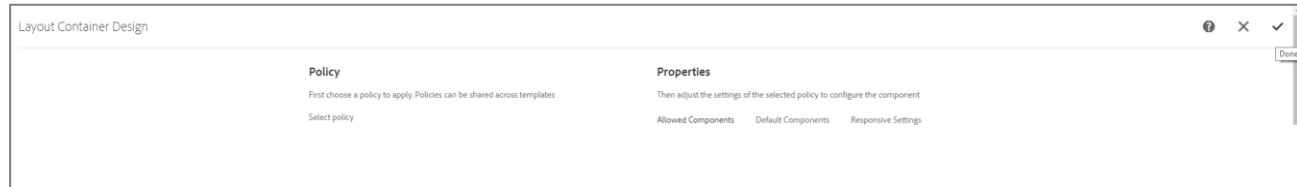
Figure 15

9. Go to any page created by updated template
10.   Select "Components" on editor panel

Figure 16

11.   Check "General" component group
12.   Drag and drop "Ooyala Video Component" on the page

To update existing Static template in AEM 6.2 do the following:

1. Open a page based on Static template
2. Switch to Design mode
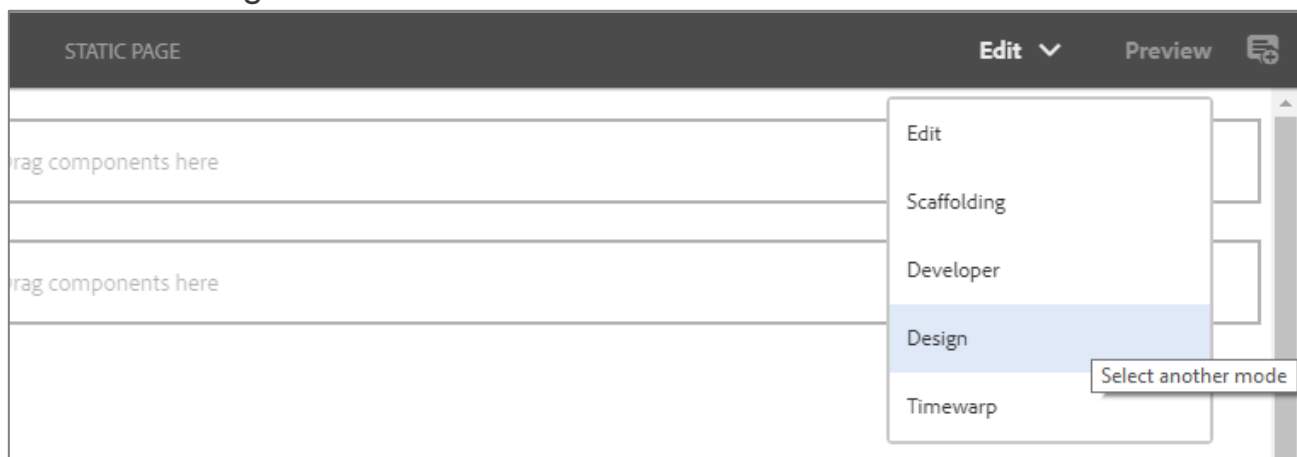
Figure 17

3. Click on "Configure" button of necessary parsys
4. On "Parsys Design" popup expand "General" components group
5. Check "Ooyala Video Component"
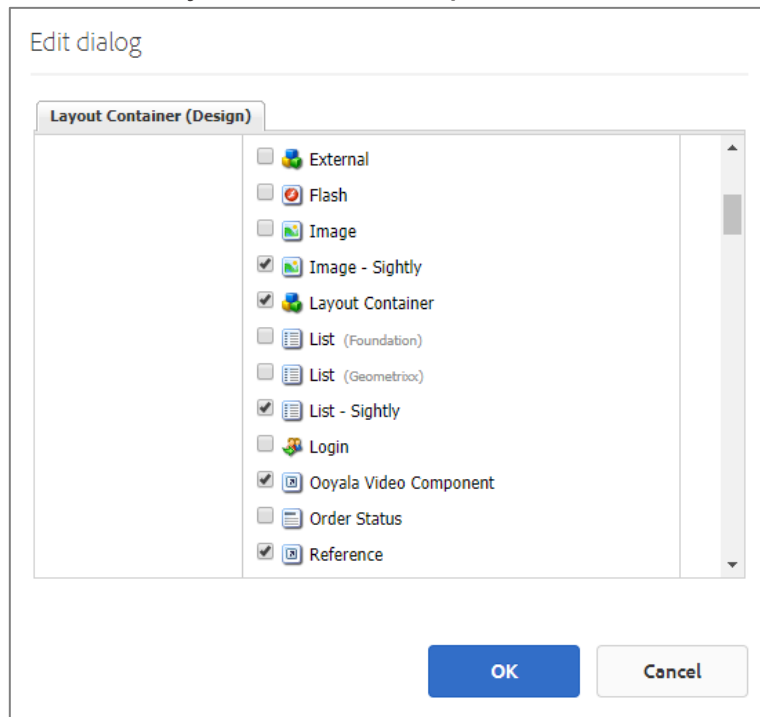
Figure 18

6. Save changes pressing "Ok" button
7. Switch to "Edit" mode

To update existing Static template in AEM 6.3 do the following:

1. Open a page based on Static template
2. Switch to Design mode
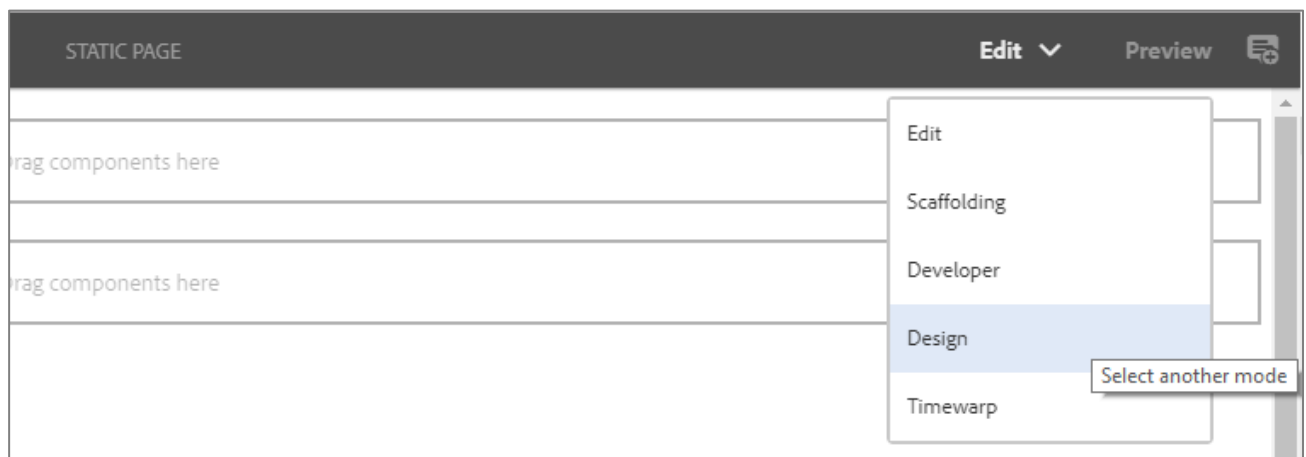
Figure 19

3. Click on "Configure" button of necessary parsys

Figure 20

4. On "Parsys Design" popup expand "General" components group
5. Check "Ooyala Video Component"
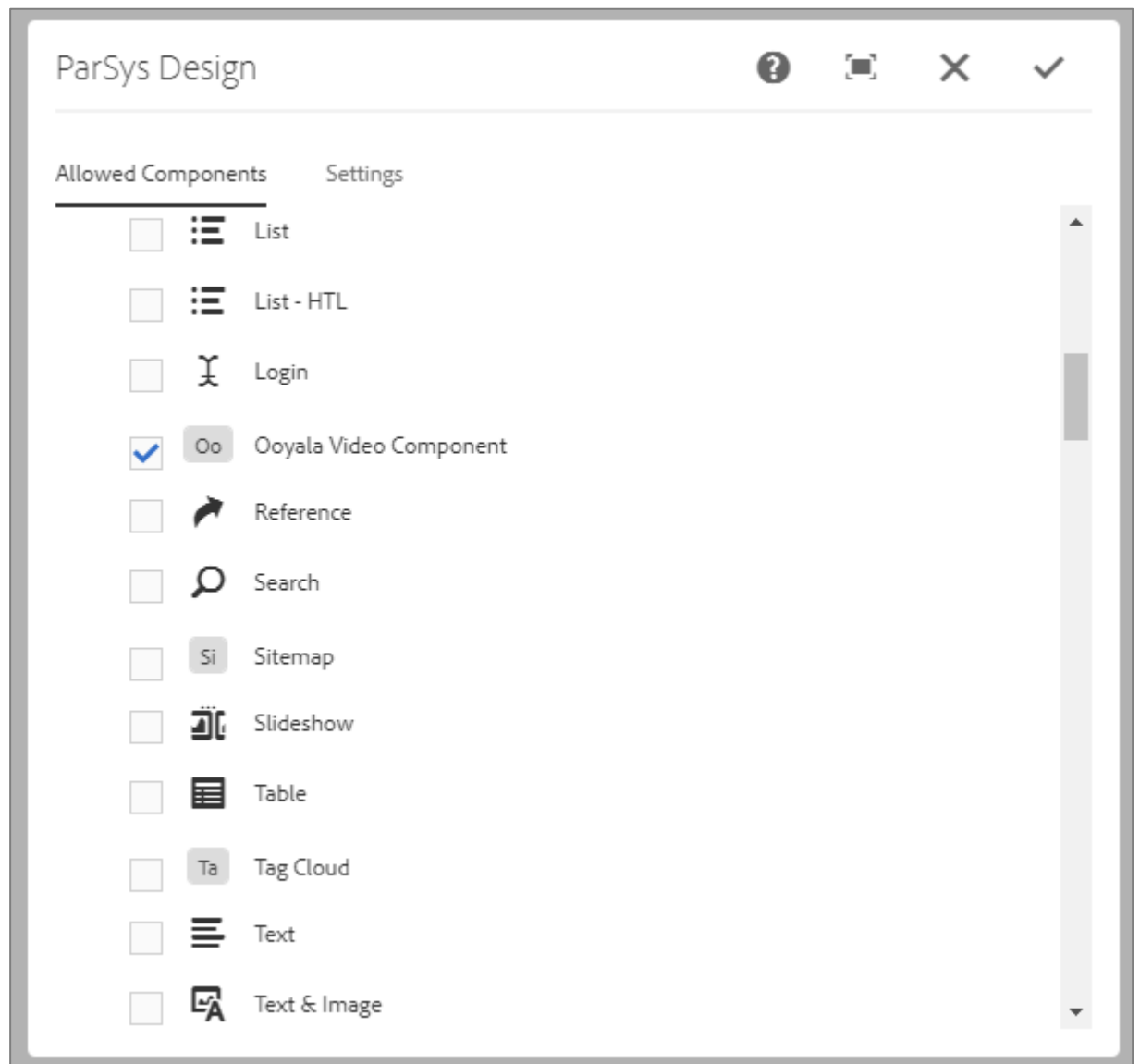
Figure 21

6. Save changes pressing "Done" button
7. Switch to "Edit" mode
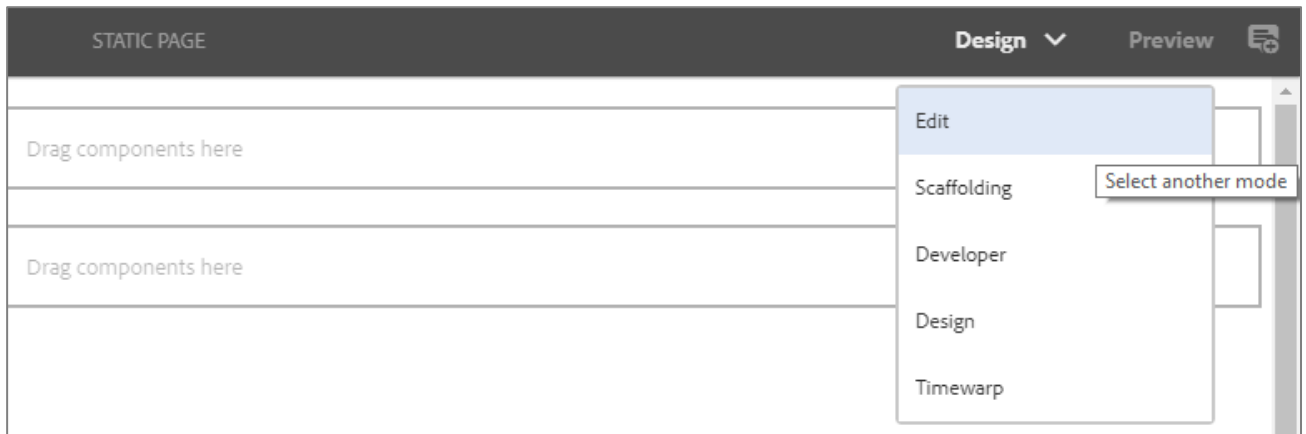
Figure 22

8. Select "Components" on editor panel

Figure 23

9. Check "General" component group
10.   Drag and drop "Ooyala Video Component" on the page

When the Ooyala video component is dropped on a page, the component renders the following message.

- Video Not Provided
- Player ID Not Provided
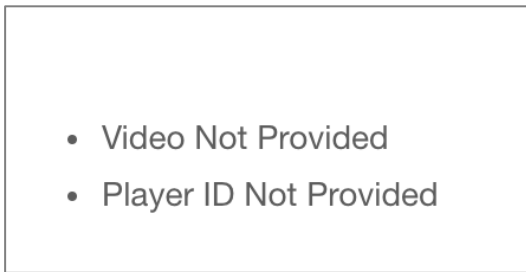
Figure 24

The next step is to specify player parameters and choose player view through component edit dialog.

'Player properties' tab allows to set width & height for video container. Video will be stretched according to values set in these fields.

Clicking 'Auto Play' checkbox will trigger automatic video playing once the page is opened.
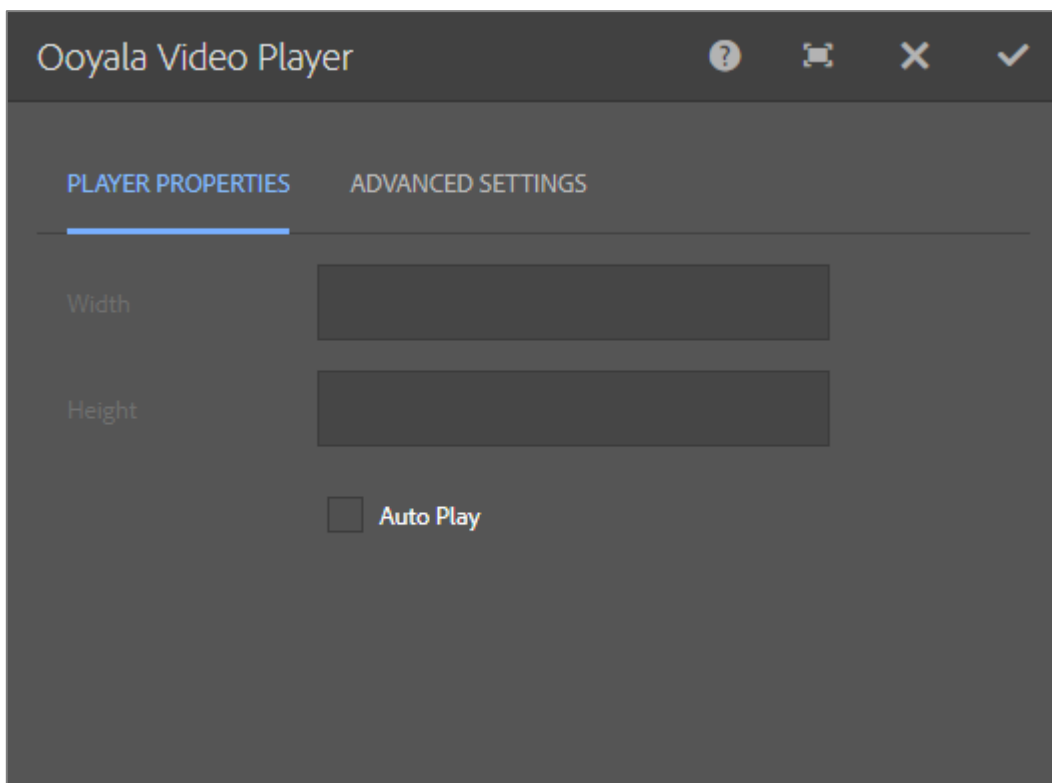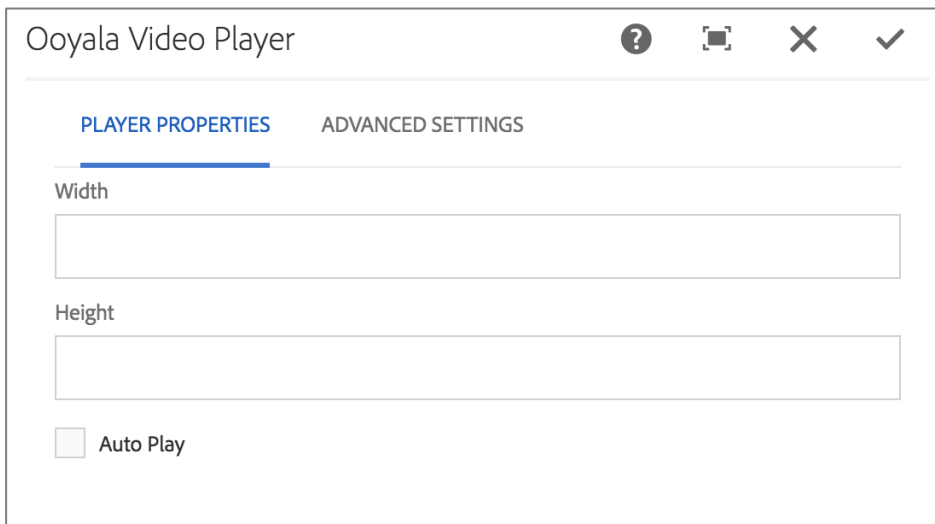
Figure 25 (AEM 6.2)

Figure 26 (AEM 6.3)

'Advanced Settings' tab contains the field for Video Embed code to be placed. Choosing video player is also obligatory – 'Choose Player' dropdown displays the list of players set in Ooyala Backlot. Player V4 is chosen as the default one.

Figure 27 (AEM 6.2)
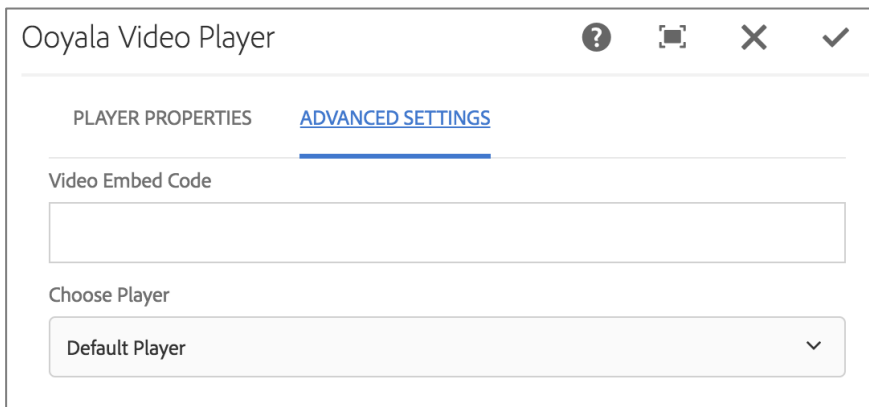
Figure 28 (AEM 6.3)

> **NOTES:**
>
> 1. *In order to view the video adjusted switch to Preview mode*
> 2. *When switching from Preview mode back to Edit mode, page reloading is required*

SEARCH OOYALA VIDEO

Assets browser can be used for searching and filtering through Ooyala videos for easy placement on AEM pages. Assets browser is available from any page in the authoring environment. In order to use the Ooyala video finder functionality, choose 'Ooyala' from selection list.
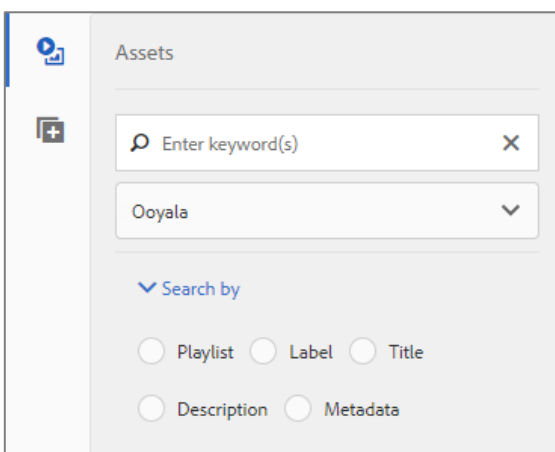
Figure 29 (AEM 6.2)
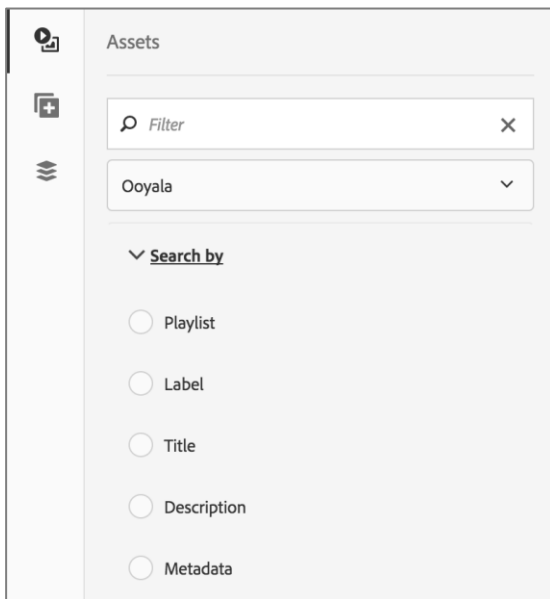
Figure 30 (AEM 6.3)

Opening 'Search by' dropdown allows five different search functionalities; search by:

- Playlist
- Label
- Title
- Description
- Metadata

Title & Description

Search by title and by description accept plain text queries and return the results that best match. Common words such as "a" and "the" are excluded from the search.

Label

Search by label allows the user to enter a label used on their Ooyala account and returns all those videos tagged with that label. The search by label requires an exact spelling match in order to function; for this reason, it provides the user with suggestions in an autocomplete box as shown.
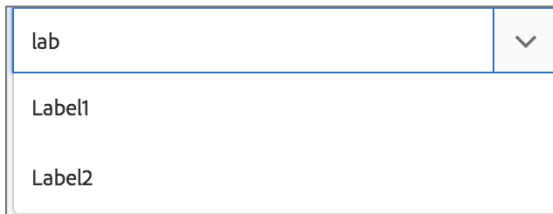
Figure 31

Metadata

Searching by metadata shows suggestions in the same manner as searching by label, based on metadata keys entered by an administrator on the Ooyala configuration screen. Searching by metadata requires both a key and a value. Choosing a key from the autocomplete prefills the search field with the key and a colon (:). Users should enter the value they wish to search for within that key after the colon, before submitting their query. Remember, it is important to make use of the autocomplete functionality to ensure your search is using the correct spelling and returning the proper results.

Playlist

In order to close in on playlists user should click 'Playlist' – this way the system will search by playlist names. Search by playlists accept plain text queries and return the results that best match.

 Playlist items are displayed in search results with static thumbnail & have 'Playlist' label as indicator.
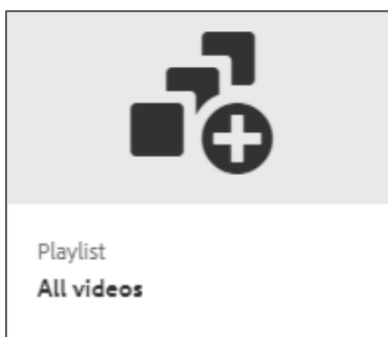
Figure 32

To add video from the search results on the page:

1. Drag the desired video item from the list
2. Drop this on top of 'Ooyala Video Component'

As a result, 'Video Embed Code' will be automatically populated.

In case with playlists dropped to the component, playlist bar will show up in the page:
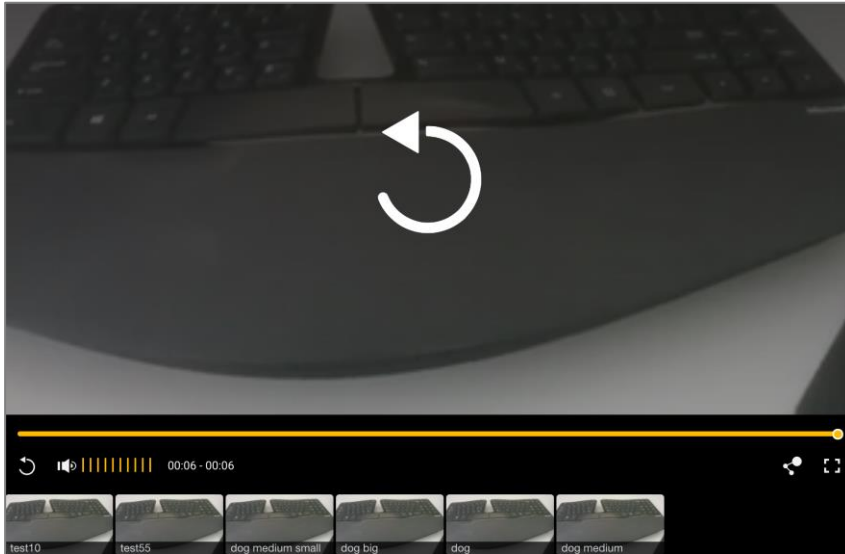
Figure 33

> **NOTE:** *Please, make sure that the player chosen has been configured to support playlists (in Ooyala Theme Builder).*

UPLOADING A VIDEO

Videos can be uploaded directly from AEM environment. On the Start Page (/aem/start) go to Tools section and select "Ooyala" panel. Open uploader page by clicking "Ooyala Uploader" on the opened panel.
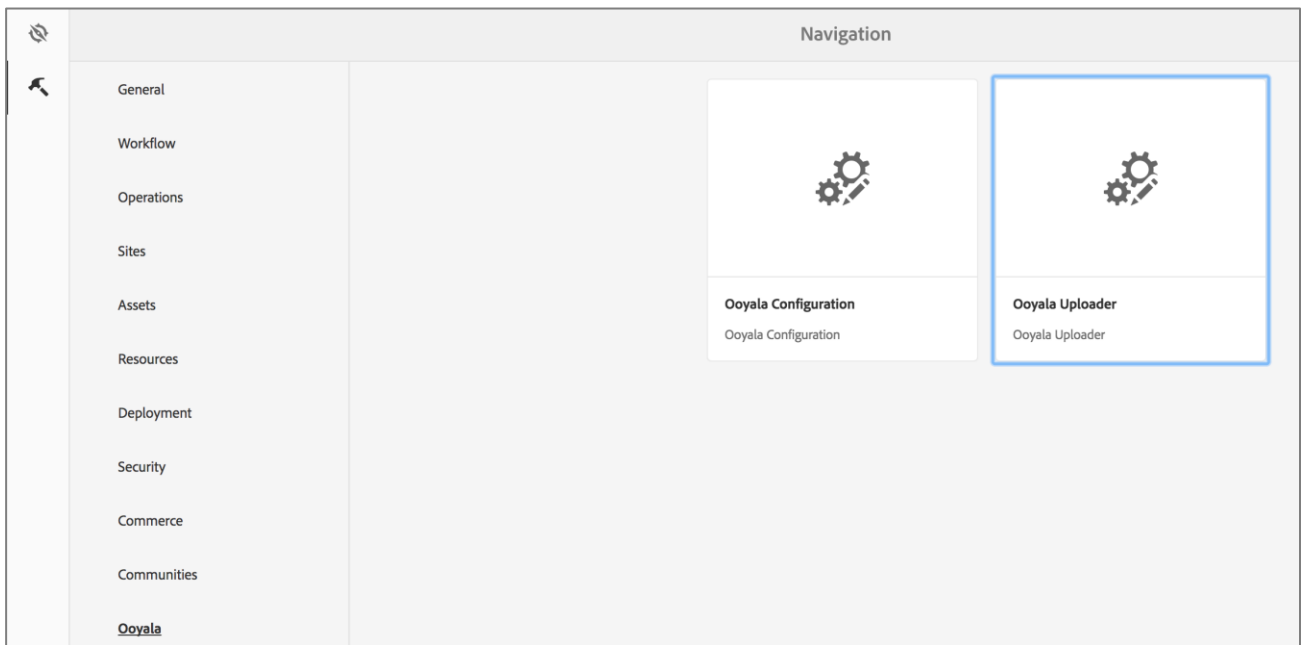
Figure 34

On the page there are three call to actions, the name and description are not required:

1. Name (text field)
2. Description (text field)
3. Select Asset (button)

The select asset button will launch a native file uploader, where the user could navigate their file system and find the video to upload.

Figure 35

Once the video has been selected and the proper name and description have been added, click the Upload button. The upload button will take the video,

and create chunks that will then be uploaded into Ooyala's and in a few minutes, users could find the video in Backlot.

Progress bar reflects the progress of video uploading.
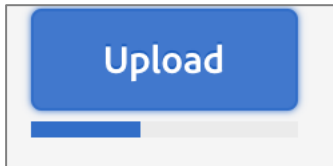
Figure 36

Once the video is uploaded Successful message will be displayed.
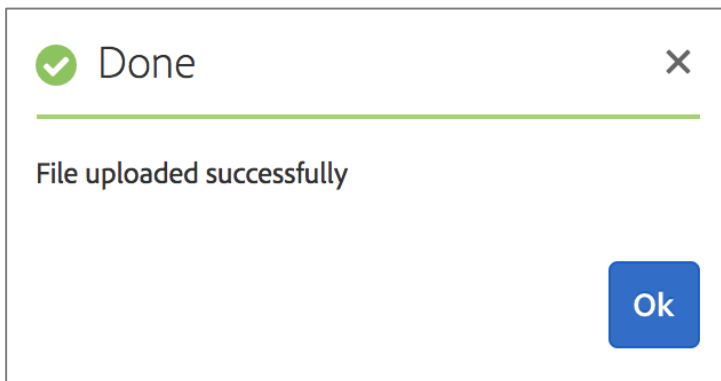
Figure 37

Then file uploaded will be in processing mode in Backlot.
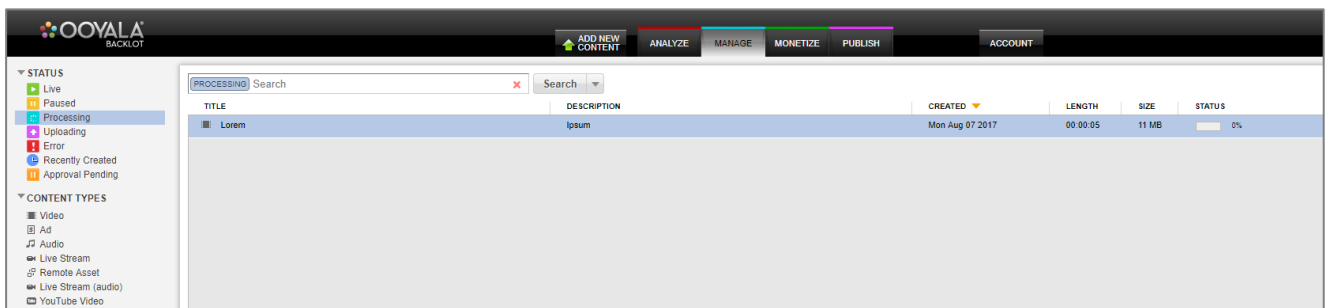
Figure 38

If upload is failed the relevant message will be shown.
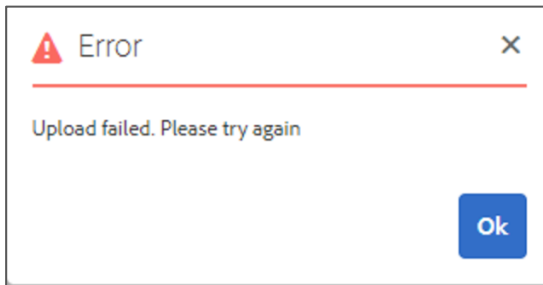
Figure 39

Validation

'Upload' button triggers validations of 'Name' field to be filled out & asset to be selected.
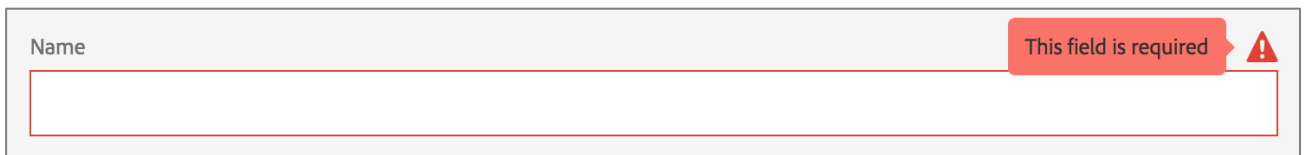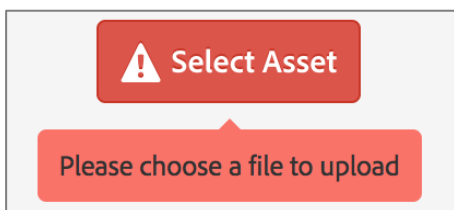
Figure 40

Figure 41

AEM allows to upload video files with extensions supported by Backlot:

wmv, avi, mov, moov, mpg, mpeg, m2t, m2v, vob, flv, mp4, mpg4, mkv, asf, m4v, m2p, 3gp, 3g2, f4v, mp3, m4a, wma, aac, mxf, wav, gxf, fcp, dv, aif, pek, and cfa.

In case of asset format is not supported by Ooyala Backlot, pop-up will be on hover 'Select Asset' button.
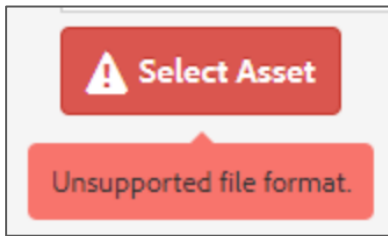
Figure 42

If API Key and/or API Secret isn't filled out 'Ooyala Uploader' page will be disabled.

Figure 43

TECHNICAL OVERVIEW

Please, refer to class diagram on the same level as current documentation.

OOYALA AEM INTEGRATION

Class	Description
OoyalaApiCredential.java	Contains one API key/secret pair.
OoyalaService.java	An Interface that provides methods to call Ooyala API (REST). Has methods such as getVideos, getLabels, etc. Methods take an OoyalaApiCredential, this allows for multiple credentials (per site/path).
OoyalaServiceImpl.java	An implementation of OoyalaService interface, OSGi service.
OoyalaClient.java	An interface that handles low-level implementation of REST calls. Uses Jersey client internally.
OoyalaClientImpl.java	An implementation of OoyalaClient interface, OSGi service.
OoyalaRequest.java	These (abstract) classes encapsulate base parameters for a Ooyala REST API request.
OoyalaQueryRequest.java	Subclasses add more specific parameters/behavior like limit, order by, etc.
OoyalaSuggestionServlet.java	Handles autocomplete/suggestion. Delegates to OoyalaService. - responds to: GET /bin/wcm/ooyala/suggestions?query=[query] - returns JSON-formatted response.

Class	Description
OoyalaPlaylistServlet.java	Handles playlist information requests from the Ooyala Video Component. Used to determine is the request parameter playlist id or single video id. Returns playlist information if request parameter is playlist id. Delegates to OoyalaService. • responds to: GET /bin/wcm/ooyala/playlist?playlistId=<playlistId> • returns JSON response from Backlot.
PlayerListDataSource.java	Matches to specific resource type, which is used in Ooyala Video component. Delegates to OoyalaService. • responds to: GET • resource type: ooyala/components/playerListDataSource
OoyalaViewHandler.java	Ooyala specific implementation, extending ViewHandler class to handle getting Backlot Video and Playlist asserts requests.
OoyalaViewQuery.java	Ooyala specific abstract implementation of ViewQuery interface. Used by OoyalaViewHandler class. Performs requests to Backlot, constructs result objects and delegates rendering of the search results.
PlaylistViewQuery.java	Playlist specific implementation of OoyalaViewQuery class. Used by OoyalaViewHandler class.

Class	Description
VideoViewQuery.java	Single video specific implementation of OoyalaViewQuery class. Used by OoyalaViewHandler class.
OoyalaViewQueryFactory.java	The factory class to get necessary extension of OoyalaViewQuery class. Used by OoyalaViewHandler class.
OoyalaAssetModel.java	The Sling Model for Backlot Video asset. Construct object by values of Hit object, which is located in SlingHttpServletRequest. Used by ooyalaPlaylistAsset and ooyalaAsset Resource types.
VideoPlayerModel.java	The Sling Model for Ooyala Video component. Construct object by dialog values. Used by ooyalaVideo component.
OoyalaConfigurationService.java	An interface that describes API for handling Ooyala Configuration.
OoyalaConfigurationServiceImpl.java	An implementation of OoyalaConfigurationService interface, OSGi service.
AbstractValidationServlet.java	An abstract class for validation servlets. Contains an API to update response and perform GET requests.
CredentialsValidationServlet.java	An extension of AbstractValidationServlet class with credentials-specific validation implementation. • responds to: GET /bin/wcm/ooyala/validation/credentials

Class	Description
	• returns JSON-formatted response.
VideoValidationServlet.java	An extension of AbstractValidationServlet class with video name-specific validation implementation. • responds to: GET /bin/wcm/ooyala/validation/video?name=<name> • returns JSON-formatted response.
SecurityHelper.java	OSGi service to distribute Service ResourceResolver based on AEM Service user with specific JCR privileges. Avoids usage of Administrative ResourceResolver and Session.
SearchType.java	Enumeration with all available asset's search types.
OoyalaClient.java	An interface that handles low-level implementation of REST calls (GET/POST/PUT).
OoyalaClientImpl.java	An implementation of OoyalaClient interface.
OoyalaPostRequest.java	Produce Ooyala REST API POST request. Extension of OoyalaRequest class.
OoyalaPutRequest.java	Produce Ooyala REST API PUT request. Extension of OoyalaRequest class.
GetAssetUploadingURL.java	An extension of OoyalaRequest class for retrieving URLs for a given asset.

Class	Description
GetLabels.java	An extension of OoyalaRequest class for retrieving labels via Ooyala's RESTful API.
GetPlayers.java	An extension of OoyalaRequest class for retrieving players via Ooyala's RESTful API.
GetPlaylistInfoRequest.java	An extension of OoyalaRequest class for retrieving playlist information via Ooyala's RESTful API.
GetPlaylistsRequest.java	An extension of OoyalaRequest class for retrieving playlists via Ooyala's RESTful API.
GetVideos.java	An extension of OoyalaRequest class for retrieving videos via Ooyala's RESTful API.

The following components are included in the plugin package:

Component	Description
ooyalaVideo (apps/ooyala/components/content/ooyalaVideo)	A Standard methodology to creating AEM components. When used on a page, It is placed on a cq:Page wherever a Video should be displayed.
ooyalaAsset (apps/ooyala/components/content/ooyalaAsset)	Specific resource type to render Backlot Video asset as Coral Card element which is used in Edit mode (editor.html)
ooyalaPlaylistAsset (apps/ooyala/components/content/ooyalaPlaylistAsset)	Specific resource type to render Backlot Playlist asset as Coral Card element which is used in Edit mode (editor.html)

Component	Description
suggestpredicate (apps/ooyala/components/content/suggestpredicate)	Ooyala specific search predicate to suggest available values when search performs by Label or Metadata.
/apps/ooyala/clientlib/common	The client library with JavaScript validation implementation and global JavaScript objects.
/apps/ooyala/clientlib/player	The client library with UI part of Ooyala Video component.
/apps/ooyala/clientlib/ooyala-backlot-sync	The client library with UI part of integration Backlot to AEM. An extension of OOTB AEM implementation to produce Ooyala specific search requests and renders the results.
/apps/ooyala/clientlib/suggestpredicate	The client library with UI part of Ooyala specific search with suggestions. Performs suggestion requests, renders the results.
/apps/ooyala/clientlib/uploader	The client library with UI part of Ooyala Uploader page.
/apps/ooyala/clientlib/configuration	The client library with UI part of Ooyala Configuration page.
/etc/ooyala/ooyala-configuration	This is a configuration page. It is used to configure the API key(s)/secret(s) in OoyalaConfigurationService.java.
/etc/ooyala/ooyala-uploader	Ooyala Uploader page. Used to upload videos directly to Backlot from AEM.

Component	Description
/etc/ooyala/synthetic-resource	A JCR resource to OoyalaViewHandler class. It needs to render Backlot response as a JCR resource.

MAVEN PROJECT STRUCTURE

Structure	Description
DEPLOY	The AEM Package available for installation
DOCUMENTATION	Documentation
CORE	The back end services
UI	The code for the component and front end.

To compile the code, a mvn clean install in the parent directory of where the code is checked out. Navigate to deploy/target directory to get a zip file to deploy into AEM 6.3 instance.

LOG FILES

AEM reports its current state into its log files, and it is often the best way to debug errors in the system:

File Name	File Location	Description
access.log	AEM_HOME/crx-quickstart/logs	Contains reports of all incoming HTTP requests

File Name	File Location	Description
error.log	AEM_HOME/crx-quickstart/logs	Contains standard log information along with any error messages that had occurred.
reques.log	AEM_HOME/crx-quickstart /logs	Contains all incoming requests/responses.