KDP events

# Passive events

Listed below are notifications fired by the KDP for INTERNAL purposes. Sending these notifications via external interface is a risk as it will very probably damage the KDP workflow. Most of these notifications are RESPONSES for loads and specific actions within the player. They are best LISTENED TO.

1. STARTUP - startUp

The first command that register the main proxys and main view mediator

2. INITIATE\_APP - initiatApp

Start the init macro commands

3. SKIN\_LOADED - skinLoaded

Dispatched when the skin is loaded

4. SKIN\_LOAD\_FAILED - skinLoadFailed

Skin load failed

5. ENTRY\_READY - entryReady

The Entry is set

6. ENTRY\_FAILED - entryFailed

Get Entry failed

7. SOURCE\_READY - sourceReady

When the source is ready we can set the media element to the media player

8. MEDIA\_READY - mediaReady

The loadable media element has been constructed and awaits loading.

9. ROOT\_RESIZE – rootResize

The KDP has completed resizing

10. MEDIA\_VIEWABLE\_CHANGE - mediaViewableChange

Used to know when OSMF Media Player is viewable. Fired when triggered by OSMF event DISPLAY\_OBJECT\_CHANGE.

11. PRE\_1\_START - pre1start - DEPRECATED

Strictly INTERNAL notification fired to trigger custom advertising plug-in action into beginning to play a pre-roll sequence.

12. POST\_1\_START - post1start - DEPRECATED

Strictly INTERNAL notification fired to trigger custom advertising plug-in action into beginning to play a post-roll sequence.

13. KDP\_READY - kdpReady

Dispatched when the KDP is Ready – meaning that the layout has completed construction and a media was loaded into the media player.

14. KDP\_EMPTY - kdpEmpty

Dispatched when the KDP layout has completed construction, but no media was loaded into the media player.

15. LAYOUT\_READY - layoutReady

Dispatched when the init macro command is done and the layout has completed contruction.

16. PLAYER\_STATE\_CHANGE - playerStateChange

Indicates the state the player has ALREADY passed to - the states are “uninitialized”, “loading”, “ready”, “buffering”, “playing”, “paused”, and “playbackError”.

Body of the notification : playerState : String

17. PLAYER\_READY - playerReady

Dispatches when player ready with content loaded into the media player.

18. PLAYER\_PAUSED - playerPaused

Sent when the player has paused the media (meaning the player is already in the “paused” state).

19. PLAYER\_PLAYED - playerPlayed

Sent when the player has started playing the entry. (meaning the player is already in the “playing” state”).

20. PLAYER\_SEEK\_START - playerSeekStart

Dispatched when the player seeking property changed to true - meaning, when the seek operation has already started.

21. PLAYER\_SEEK\_END - playerSeekEnd

Dispatched when the player seeking property changed to false – meaning the seek operation has already ended.

22. PLAYER\_PLAY\_END - playerPlayEnd

Dispatched when the media has reached the end - meaning currentTime == duration

23. PLAYER\_UPDATE\_PLAYHEAD - playerUpdatePlayhead

Dispatched every time the entry time has updated - occurs at a regular interval of about 0.3 of a second.

Body of the notification : currentTime : Number;

24. VOLUME\_CHANGED - volumeChanged

Fired when the volume of the media has been changed in the OSMMF media player level (meaning the volume has actually changed).

25. BYTES\_DOWNLOADED\_CHANGE - bytesDownloadedChange

Notify the current and the previous value of bytesDownloaded

26. BYTES\_TOTAL\_CHANGE - bytesTotalChange

Dispatched by the player when the value of the property "bytesTotal" has changed.

27. BUFFER\_PROGRESS - bufferProgress

Dispatched when the player has buffered the limit of buffering time specified for it.

28. BUFFER\_CHANGE - bufferChange

Dispatched when the player start or stop buffering

29. DURATION\_CHANGE - durationChange

The player dispatches this event when the duration of the media loaded into the OSMF media player has changed (for instance, when a new media is loaded).

30. HAS\_CLOSED\_FULL\_SCREEN - hasCloseFullScreen

The player (already) left full-screen mode.

31. HAS\_OPENED\_FULL\_SCREEN - hasOpenedFullScreen

The player has (already) gone into full-screen mode.

32. SWITCHING\_CHANGE - switchingChange

Dispatched when the KDP has started or finished switching to a new bitrate flavor.

33. SCRUBBER\_DRAG\_START - scrubberDragStart

The user started dragging the scrubber.

34. SCRUBBER\_DRAG\_END - scrubberDragEnd

The user stopped dragging the scrubber.

35. PLAYBACK\_COMPLETE - fired when a media playing in the player (not an external swf) has finished playback.

36. SEQUENCE\_ITEM\_PLAY\_END - fired when a sequence plugin has finished playback (including any sub-sequence it had if any). This notification is also fired for external swf-s.

37. PRE\_SEQUENCE\_COMPLETE - fired when the pre-sequence has finished playback.

38. POST\_SEQUENCE\_COMPLETE - fired when the post-sequence has finished playback.

39. READY\_TO\_PLAY - readyToPlay

The KDP has a valid entry object which is ready to be loaded as a media element and played. This notification is fired repeatedly at the end each CHANGE\_MEDIA process.

40. READY\_TO\_LOAD - readyToLoad

The KDP either does not have an entryID and has not loaded an entry object, or is prevented from playing the loaded entry (the entry in question is under access control, or has an unplayable moderation

status).

41. MEDIA\_LOADED - mediaLoaded

The KDP successfully loaded a media element into the OSMF media player and the element is ready to be played.

42. METADATA\_RECEIVED - metadataReceived

The metadata related to the entry has been retrieved.

43. CHANGE\_MEDIA\_PROCESS\_STARTED - changeMediaProcessStarted

Because the CHANGE\_MEDIA notification is a trigger for a pureMVC command which delays all mediators handling the notification until the command is finished, this notification

is fired from the first command in the CHANGE\_MEDIA macro-command, if a mediator must act as soon as the process started.

44. SWITCHING\_CHANGE\_STARTED - switchingChangeStarted

Notification dispatched when the player has started switching to a different dynamic bitrate.

Body of the notification - {newIndex: [the index of the bitrate the player started switching to. If auto, send -1],newBitrate: [the bitrate the player started switching to. If auto, sned null]}

45. SWITCHING\_CHANGE\_COMPLETE - switchingChangeComplete

Notification dispatched when the player has finished switching to a different dynamic bitrate.

Body of the notification - {currentIndex:[the index of the bitrate that the player finished switching to],currentBitrate: [the bitrate the player finished switching to]}

# Advertising related events (mainly for tracking purposes).

39. BUMPER\_STARTED = "bumperStarted"

notification sent when a bumper video starts to play.

40. BUMPER\_CLICKED = "bumperClicked"

notification sent when a bumper video is clicked.

41. AD\_START = "adStart"

notification sent when an ad shows.

this notification will have a data object {timeSlot:preroll/postroll/midroll/overlay}

42. AD\_CLICK = "adClick"

notification sent when an ad is clicked.

this notification will have a data object {timeSlot:preroll/postroll/midroll/overlay}

43. AD\_END = "adEnd"

notification sent when a video ad ends.

this notification will have a data object {timeSlot:preroll/postroll/midroll/overlay}

44. FIRST\_QUARTILE\_OF\_AD = "firstQuartileOfAd"

notification sent when a video ad reaches 25%

this notification will have a data object {timeSlot:preroll/postroll/midroll/overlay}

45. MID\_OF\_AD = "midOfAd"

notification sent when a video ad reaches 50%

this notification will have a data object {timeSlot:preroll/postroll/midroll/overlay}

46. THIRD\_QUARTILE\_OF\_AD = "ThirdQuartileOfAd"

notification sent when a video ad reaches 75%

this notification will have a data object {timeSlot:preroll/postroll/midroll/overlay}

# Active events

The following are TRIGGER events, which means that they can be fired by the external interface to trigger the required result.

1. CHANGE\_MEDIA - changeMedia

Start the init of change media macro commands

Body of notification is an object of the following structure :

{entryId : String, flavorId : String, preferedFlavorBR (the selected bitrate for the entry) : Number}

2. CLEAN\_MEDIA - cleanMedia

Cleans the media in case of change media and entryId== -1

3. DO\_PAUSE - doPause

Command the player to pause;

4. DO\_PLAY - doPlay

Command the player to play;

5. DO\_STOP - doStop

Do stop command to the kplayer. Pause and move the playhead to 0

6. DO\_SEEK - doSeek

Do seek command to the kplayer. Additional data - number

7. DO\_SWITCH - doSwitch

Do switch command for switching manual switching between mbr streams within an rtmp dynamic stream resource

Body of the notification is:

preferedFlavorBitrate : Number and carries the prefered bitrate to switch to.

8. OPEN\_FULL\_SCREEN - openFullScreen

open full screen

9. CLOSE\_FULL\_SCREEN - closeFullScreen

close full screen command

10. CHANGE\_VOLUME - changeVolume

change volume. Body of the notification - volume value (0 to 1)

11. ENABLE\_GUI - enableGui

enable/disable gui. Addition data - Object{guiEnabled : Boolean,enableType : String}

12. ALERT - alert

Pop up an alert. arguments: message, title, buttons, callbackFunction, iconClass

message - string containing the message to display in the alert

title - string containing the title of the alert

buttons - array of strings for the alert button labels

callbackFunction - the function which handles an alert button being clicked.

iconClass - string containing the name of the export class of the icon that the alert should display (should exist in the skin.swf file).

13. REMOVE\_ALERTS - removeAlerts

Removes all alerts currently displayed by the player.