Webcast - HLS based webcast

Scope

This page purpose is to document all changes and modifications to the existing Webcast THEME (WMS based), including states, state machine, endpoints and flows.

Webcast state machine (transition and actions)

From\To	NotFound	Ready	InProgress	Paused	Stoppped	StreamingError	Error
NotFound		S					S
Ready			R	R	R		R
InProgress				R	R	R	R
Paused			R		R		R
Stopped							
StreamingError			R	R	R		R
Error							

Legend:

- R - Recorder transition.
- S - Server transition.

Folders (In, Out)

- Lamda source files folder (Temp bucket)
    - `TempBucket/WebcastsFiles/[WebcastId]/`
- Lamda destination files folder (Recording bucket)
    - http://tegrity-nonprod.s3.amazonaws.com/tlc/dev/Tenants/[CustomerId]/Recordings/[CourseId]/[WebcastId]/Class/Projector/Screen
    - http://tegrity-nonprod.s3.amazonaws.com/tlc/dev/Tenants/[CustomerId]/Recordings/[CourseId]/[WebcastId]/Class/Projector/Screen.m3u8

APIs

```
GET /api/webcast/[WEBCAST_ID]/details
```

Response sample for webcast in **NotFound** state:

```
{
  "id": "9d1bf276-110a-4d41-b95c-49c1ed62067f",
  "courseId": "",
  "recordedBy": "",
  "title": "",
  "state": "NotFound",
  "tempBucket": null,
  "mediaStreams": []
}
```

Response sample for webcast in any state other than **NotFound** state:

```
{
  "id": "9d1bf276-110a-4d41-b95c-49c1ed62067f",
  "courseId": "2596fabf-cb8f-4e6d-9422-58761e2e84d1",
  "recordedBy": "2596fabf-cb8f-4e6d-9422-58761e2e84d1",
  "title": "Test webcast",
  "state": "Ready",
  "tempBucket":
  {
 "accessToken":
 {
     "accessKey": "XXXX",
     "secretKey": "YYYYY",
     "sessionToken": "ZZZZ"
 },
    "blobStorageIdentifier":
    {
      "regionName":"west-2",
      "containerName":"[TempBucket]", //e.g. "tegrity-nonprod"
      "blobName":"[WebcastsFilesFolder]/[WebcastId]" //e.g.
"tlc/qabr/Temp/WebcastFiles/29b83601-da01-47a6-888b-21f02354692e"
 }
   },
  "manifestUpdateURL" : "https://[URL-according-to-stage-from-config]",
  "mediaStreams":
   [
    {
   "streamId": "[StreamId]",
      "streamType": "AudioVideo",
      "timer": true,
      "width": 1024,
      "height": 768,
      "vCodec": "{31435657-0000-0010-8000-00AA00389B71}",
      "aCodec": "{14537685-0000-0010-8000-00AA0F389D72}",
   "blobStorageIdentifier":
     {
     "regionName":"west-2",
        "containerName":"[RecordingBucket]", //e.g. "tegrity-nonprod"

"blobName":"tlc/dev/Tenants/[CustomerId]/Recordings/[CourseId]/[WebcastId]
/Class/[ControlId]/[StreamId].m3u8" //e.g.
"tlc/dev/Tenants/cd8ffbf3-1fbc-430c-a8c9-816132e44355/Recordings/29b83601-
da01-47a6-888b-21f02354692e/9d1bf276-110a-4d41-b95c-49c1ed62067f/Class/Pro
jector/Screen.m3u8"
      }
    }
   ]
}
```

The recorder will upload the per-stream segmented mp4 files to the location indicated by the server in the blobStorageIdentifier property, and append the StreamId value, i.e.:

"WebcastsFiles/[WebcastId]/[StreamId]/[StreamId]_[SegmentIndex].mp4",
for example:
"WebcastsFiles/29b83601-da01-47a6-888b-21f02354692e/Screen/Screen_00001.mp4"

Flows

Server actions upon call to **POST /api/webcast/create**

1. Sever creates rec in Mongo DB - state = **NotFound**
2. Temp bucket: create temp credentials for Temp\WebcastFiles\[WebcastId]
3. Recording bucket: after copying skeleton from template , update the Screen.xml
4. TargetConfig.js: Create TargetConfig.js files per stream, at the Temp bucket.
5. Change webcast state to **Ready**

Example for Screen.xml:

Sample Screen.xml

```
<?xml version="1.0" encoding="UTF-8">
<DATA id="Screen" time="0" dur="172800000" type="VideoAudio" stream="true">
  <SGMTS>
    <SGMT dur="172800000" time="0" type="VideoAudio" inx="1" height="1080"
width="1920" fmt="HLS">
      <SMPL inx="1" src="[StreamId].m3u8" time="0" dur="172800000"
timer="true"/> <!-- e.g. src="Screen.m3u8" -->
    </SGMT>
  </SGMTS>
  <PREL>
    <ITEM src="[StreamId].m3u8" type="VideoAudio" fmt="HLS"
vcodec="[VCodec]" acodec="[ACodec]"/> <!-- e.g. src="Screen.m3u8" -->
  </PREL>
</DATA>
```

Changes in the Recording's metadata to support HLS playback

Session.xml (Session.js)

```
<PROP name="Duration" value="172800000"/><!-- 48 Hours -->
<PROP name="MacSupport" value="HLS"/><!-- Optional values are
[WM|QT|MP4|HLS] -->
<PROP name="RecordingMode" value="WEBCAST"/> <!-- Optional values are
[CLASS|PROCTOR|WEBCAST], default value is "CLASS" -->
```

Stream.xml (Stream.js)

```
<SGMT inx="1" type="VideoAudio" fmt="HLS" fmt1="MPEG-DASH" time="0"
dur="172800000">
```

Configuration Files

WebcastState.wbcst

This file keeps the scheduling information for the manifest lambda.

The manifest lambda is triggered by a write event of this file.

The recorder creates this file **once**, before uploading any fragments, at:

`"[WebcastsFilesFolder]/[WebcastId]/[streamId]/WebcastState.wbcst"`

The initial contents of the file, as created by the recorder, is an empty json object (i.e. "{}")

TargetConfig.js

TargetConfig.js describes the location where the lambda should create the webcast stream.
A TargetConfig.js should be created for each of the streams that the recorder requests the server to generate.
The server will create a TargetConfig.js at:

`"[WebcastsFilesFolder]/[WebcastId]/[streamId]/TargetConfig.js"`

for example:

`"tlc/qabr/Temp/WebcastFiles/29b83601-da01-47a6-888b-21f02354692e/Screen/TargetConfig.js"`

TargetConfig.js will be generated by the server, during the handling the "CreateWebcast" request, and before the webcast state changes to "Ready".

Example TargetConfig.js:

```
{
 "blobStorageIdentifier":
   {
  "regionName":"west-2",
        "containerName":"[RecordingBucket]", // e.g. "tegrity-nonprod"

"blobName":"tlc/dev/Tenants/[CustomerId]/Recordings/[CourseId]/[WebcastId]
/Class/[ControlId]/[StreamId].m3u8" //e.g.
"tlc/dev/Tenants/cd8ffbf3-1fbc-430c-a8c9-816132e44355/Recordings/29b83601-
da01-47a6-888b-21f02354692e/9d1bf276-110a-4d41-b95c-49c1ed62067f/Class/Pro
jector/Screen.m3u8"
    }
}
```

MediaConfig.js

MediaConfig.js contains information about media segments required by the webcast lambda.
MediaConfig.js will be created by the Recorder, and uploaded before any of the mp4 segments.
MediaConfig.js is created per-stream, at:

`"[WebcastsFilesFolder]/[WebcastId]/[streamId]/MediaConfig.js"`

for example:

`"tlc/qabr/Temp/WebcastFiles/29b83601-da01-47a6-888b-21f02354692e/Screen/MediaConfig.js"`

Example MediaConfig.js:

```
{
"targetDuration":10 // This value affects the m3u8 header
#EXT-X-TARGETDURATION:
}
```

Clean up

Delete temp bucket key : TempBucket\WebcastFiles\[WebcastId]