v1.1b3 | Aug 18, 2013

VAST Library Sample

Using VAST with your own Roku channel

There are two main steps to integrating VAST preroll ads into your channel.

First, you must build a content-meta-data structure defining the preroll ad and assign it to the *preroll* attribute of your main video's content-meta-data structure. This can be accomplished with a few simple lines of BrightScript. Given a content-meta-data structure called *video* that defines your main content, the following code will construct a preroll ad from a chunk of valid VAST and assign it to that piece of content. This example assumes that the first ad contained in your VAST is a video ad and is intended to be a preroll for the content.

```
vast = NWM_VAST()
vast.Parse(vastString)
if vast.ads.Count() > 0 and ads[0].video <> invalid
  video.preroll = ads[0].video
end if
```

Once you have constructed the content-meta-data structure for your preroll ad, you are ready to play your content. in order for the preroll ad to be displayed and any associated tracking events to fire correctly, you must play your content using a modified video screen. Given the content we just defined in a content-meta-data structure called *video*, the following line of code will initiate the playback experience, displaying the preroll ad, immediately followed by the main content:

```
PlayVideo(video)
```

Notes

This library supports both the <impression> tag and the <tracking> tag for defining tracking events that should be fired during playback of the video. A tracking event specified using the <impression> tag will behave the same as an event specified using <tracking event="start">.

Proper timing of tracking events relies on the accuracy of the value of the <duration> tag. If a creative has an incorrect or missing <duration> tag, tracking events for that creative will not fire at the correct times.

Understanding the code

NWM_VAST.brs
This file contains a BrightScript class partially implementing the VAST 2.0.1 standard.

NWM_VAST()
Constructor

NWM_VAST::Parse(raw, returnUnsupportedVideo = false, normalizeURLs = false)

- **raw**: a string containing a valid VAST response
- **returnUnsupportedVideo**: boolean telling the library whether it should return videos whose MIME type is not video/mp4. Default: *false*
- **normalizeURLs** : boolean telling the library whether it should attempt to normalize tracking URLs. This may help with situations where your ad provider returns poorly encoded tracking URLs in their VAST. Default: *false*

Parses the string stored in *raw* and constructs collection of extended content-meta-data objects representing the ads defined in the VAST. The resulting array of ads is stored in the object's *ads* attribute.

NWM_VAST::GetPrerollFromURL(url)

- **url**: the absolute URL to a chunk of VAST XML

***GetPrerollFromURL* is deprecated, but has been left in the library for backward compatibility**

This function will fetch a chunk of VAST XML from the specified URL, parse it, and use it to construct an extended content-meta-data object representing a preroll video.

The function will use the first linear creative found in the VAST response which contains at least one <mediaFile> whose type has a supported value. By default, supported values are "video/mp4" and "video/x-mp4".

VAST_VideoScreen.brs
This file implements a VAST/preroll enabled video screen capable of playing a preroll video and making any asynchronous tracking calls during playback of the preroll video before beginning playback of the main content item.

PlayVideo(video)

- **video**: a content-meta-data object, as defined in the Roku *Component Reference* defining the main content to be played. This parameter should have an additional attribute called "preroll" whose value is the content-meta-data object representing the preroll video which was built by a previous call to NWM_VAST::GetPrerollFromURL

This function is the main entry point for the video experience and is the only function that should ever be called directly.