SILVERLIGHT INTEGRATION

Introduction

This document describes steps for integrating Microsoft Silverlight Players with Nielsen's Combined Beacon library.

This document is a suggested approach to implementation. It is not mandated and you may choose your own way of managing and passing meta-data to the gg() class.

For more detail on the gg() class, see the "Combined Beacon JS API Integration" document. For Silverlight, we use the "native" Javascript mode as described in that document.

ggslXXX.js (where XXX is the version number) is our Event Handler library for Silverlight – simply including that typically automates data capture for core events like LoadVideo, Play and Pause.

Microsoft Silverlight Analytics Framework: This Nielsen beacon has also been incorporated into the Microsoft Silverlight Analytics Framework (SAF) project. For more detail see http://msaf.codeplex.com/. You will still require the Nielsen JavaScript beacon as described in step 3 below. Step 1 is not required if using the SAF. **Important caveat**: The current version in the SAF does not support stand alone desktop Silverlight variants. This is because this current version is reliant on our core JavaScript beacon rather than a native C# implementation.

Microsoft Silverlight 4: The current integration approach is based on a JavaScript interface not a Silverlight 4 native C# interface. Therefore, at this time, you are required to use a JavaScript to C# bridge in order to expose your Silverlight 4 events to the Nielsen JavaScript API.

Prerequisites and Assumptions

Knowledge of JavaScript and Silverlight 1.0, 2.0, 3.0.

The document "Combined Beacon JS API Integration" is prerequisite reading if not implementing via the Silverlight Analytics Framework.

Implementation

The following steps are required:

1. Include the Silverlight Event handler library like this:

```
<script type="text/javascript"
    src=" http://secure-XX.imrworldwide.com/novms/js/2/ggslXXX.js">
</script>
```

Note 1: "XX" is the Nielsen supplied country code for your region. For example, it is "us" for customers in USA.

Note 2: "XXX" is the version number of the JS specific Silverlight handler.

2. Initialize clientid and other parameters like this (these values will be supplied to you by Nielsen):

```
<script type="text/javascript">
        var _nolggGlobalParams = {
        clientid: “ my Nielsen assigned client id ”,
        vcid: “ my Nielsen assigned vc id ”,
        cisuffix: ““,
        sfcode: "Nielsen assigned data node “,
        prod: " Nielsen assigned product code "
};
</script>
```

3. Include and initialize Combined Beacon Javascript library for Native mode:

```
<script type="text/javascript"
   src=" http://secure-XX.imrworldwide.com/novms/js/2/ggcmbXXX.js">
</script>
```

Note 1: “XX” is the Nielsen supplied country code for your region. For example, it is “us” for customers in USA.

Note 2: “XXX” is the latest version number of the general JS beacon.

4. Complete the initialization by including the following lines:

```
<script>

        var canUseSWF = false;  // forces native mode
        var uid = 0;
        var gg1 = new gg();
        gg1.ggInitialize(_nolggGlobalParams,uid,canUseSWF);

</script>
```

Many Silverlight implementations use XAML to customize user experience. Follow the steps below if your implementation uses XAML.

5. Add “{mediaName:'media1', gg:gg1}” to userContext parameter in Silverlight createObject call. Also, add “onLoad:bindEvents” to the events parameter. For example, when this step is done, it may look like this:

```
Silverlight.createObject(
       "myxaml.xaml",      // example .xaml
        parentElement,
        "myControlID",
        {
                width:'320',
                height:'300',
                inplaceInstallPrompt:false,
                background:'#D6D6D6',
                isWindowless:'false',
                framerate:'24',
                version:'1.0'
        },
            {
                        onError:null,

                        onLoad:bindEvents         // add this line
            }, null,
            {mediaName:'media1', gg:gg1}   // add this line
);
```

6. Myxaml.xaml should include a mediaElement with the value of mediaName above. For example, you may include a line like:

```
<MediaElement x:Name="media1" Source="video.wmv" Width="320" Height="240”>
```

Passing Video Meta Data

If you supply an ASX playlist, you can supply videoinfo metadata for LoadVideo and LoadAndPlayVideo events. As described in the API document, these include <category>, <subcategory>, <title> and <uurl>. Additionally, <contenttype> can be specified to differentiate content from ads – allowable values for this are content, preroll, midroll or postroll. Integration code automatically picks up these values from the playlist.

Here’s an example .ASX playlist:

```
<ASX version = "3.0">
     <ENTRY>
           <TITLE>Test title 1</TITLE>
           <REF HREF = "video.wmv" />
            <PARAM NAME="uurl" VALUE="video.wmv" />
            <PARAM NAME="category" VALUE="cat1" />
            <PARAM NAME="subcategory" VALUE="subcat1" />
            <PARAM NAME="contenttype" VALUE="content" />
     </ENTRY>
     <ENTRY>
           <TITLE>Test Ad title 1</TITLE>
           <REF HREF = "video2.wmv" />
            <PARAM NAME="uurl" VALUE="video2.wmv" />
            <PARAM NAME="category" VALUE="adcat1" />
            <PARAM NAME="subcategory" VALUE="adsubcat1" />
            <PARAM NAME="contenttype" VALUE="postroll" />
    </ENTRY>
</ASX>
```