Channel Connect

Introduction

The Channel Connect Flash component connects a rich media unit with an HTML client which can be loaded into a desktop web browser window or into a mobile web browser.

NOTE: This component can be used with any Rich Media format.

Build & Delivery

Since you're going to use an internal version of the Channel Connect solution, no public MXP package is available. Please ask your DoubleClick Rich Media contact for the latest release of the component.

NOTE: The functionality of this component will only be fully testable when run inside DoubleClick Studio.

Steps - Flash Creative

To initialize and utilise the Channel Connect functionality within your creative, follow these steps:

1. Open the template you should have been provided with and make sure a *ChannelConnect Library* symbol is available in your FLA library.
2. It is not mandatory to have a component instance on the timeline, so you can proceed using just the ActionScript API and instantiate the object via code.
3. Open Actions Window on first frame and initialize the Studio Enabler as usual.
4. Make sure to include this boilerplate code in order to properly initialize the component:

```
import com.google.ads.studio.innovation.channelconnect.proxy.ChannelConnect;
var channelConnect = ChannelConnect.getInstance();
```

```
channelConnect.addEventListener("channelClose", channelCloseHandler);
channelConnect.addEventListener("channelOpen", channelOpenHandler);
channelConnect.addEventListener("channelError", channelErrorHandler);
channelConnect.addEventListener("channelMessage", channelMessageHandler);
channelConnect.addEventListener("channelReady", channelReadyHandler);
channelConnect.addEventListener("channelLoad", channelLoadHandler);
addChild(channelConnect);
```

It's very important to understand the events sequence dispatched by the component during the connection life cycle:

1. `"channelLoad"`: dispatched when all JavaScript libraries have been completely loaded. It is now possible to request a pair of channels to the Google App Engine service using the `"requestChannels"` method.
   NOTE: The `"requestChannels"` method accepts an optional parameter, the secret key. The secret key is need to access the service in production and will be provided by DoubleClick team. If no secret key is provided, the creative will connect to the development service with limited capabilities in terms of number of access/day.

2. `"channelReady"`: dispatched when the channels are available for the client. It is now possible to connect to the channels using the `"connectToChannel"` method.

3. `"channelOpen"`: dispatched when the connection to the channel has been successfully established. It is now possible to open the mobile client and let it connect to the available channel.

After this sequence of three events have been completed, it will be possible to:

> Implement the method of choice to open the companion client to connect to (e.g. QR code, short url, etc)
It is very important to provide the channel session id to the secondary unit, so please be sure to implement a proper way to pass the `channelConnect.proxy.getSessionId()` variable (e.g query string parameters)

> Send messages to the possible connected companion by calling the method

```
channelConnect.proxy.sendMessage({message: "Google Rocks!!!", name: "DoubleClick"})
```

The input parameter for this method can be any serializable type (e.g string, Object)

> Handle an incoming message - from the connected companion - using the appropriate event handler as described in step 4. The events to handle will be in the format

```
event
    |
    type
    |
    data
        |
        data (actual message)
```

The `event.data.data` property contains the message itself.

- Purposely close the channel connection by calling the method

```
channelConnect.proxy.closeChannel()
```

- Appropriately handle possible error (`channelError`) and/or disconnection (`channelClose`) events. Once a `channelClose` event is received it won't be possible so send over messages to a possible connected companion, so it's mandatory to establish a new connection before any further sending attempt.

NOTE: The component public API is exposed through its proxy member, i.e.:

```
channelConnect.proxy.sendMessage({})
channelConnect.proxy.getSessionId()
channelConnect.proxy.requestChannels(secretKey:String = null)
channelConnect.proxy.connectToChannel()
channelConnect.proxy.closeChannel()
```

Steps - HTML Companion

The setup in the HTML companion is pretty much the same as described in the Flash Creative section. The important difference to take care of is in the call to the ChannelConnect class constructor: the HTML companion is supposed not to be the one who establishes the channel connection, but connects to an already active connection.

1. Include the following external JavaScript file in your HTML code:

```
<script
src="http://www.gstatic.com/doubleclick/studio/innovation/channelconnect/channel_connect_v2.js">
</script>
```

2. Each connection is identified by an id which - through ActionScript public API - can be read as `channelConnect.proxy.getSessionId()`. This value should be passed as input parameter constructor in the JavaScript code of the HTML companion:

```
var channelConnect = studio.innovation.channels.ChannelConnect.getInstance();
channelConnect.addEventListener(studio.innovation.ChannelEvent.CHANNEL_READY, channelReadyHandler);
// Implement all the event listeners...

// Get the needed parameter, e.g. reading it from the query string parameters.
function getParameter(name) {
  // TBI: read the parameter from the querystring.
```

```
}

var sessionId = getParameter('sessionId');
channelConnect.getMobileChannel(sessionId);
```

NOTE: Be sure to initialize the component after the external JavaScript files have been completely downloaded - listening to `<body onload="loadHandler();" />` is a suitable approach.

Make sure to use the ChannelMessage class to send messages when calling the `channelConnect.sendMessage(message)` method - this is transparent in ActionScript since the Flash component itself takes care of wrapping the raw message into a ChannelMessage object; on the other side it is mandatory to do this explicitly when coding JavaScript.

ChannelConnect AS3 API Reference

You can find detailed AS3 API reference her .

Limitations And Constrains

The solution forces a set of constraints in its implementation:

1. The first call to the `requestChannels` method (in order to get the session id and possibly display the correct url to the mobile companion, ie by loading a QR code) **must be executed on explicit interaction** by the user, through a CTA button. This means that all creatives calling the above method on impression or after just `StudioEvent.INTERACTION` event (ie mouse hovering on the creative) **will be QA rejected**.
The component also checks the `HtmlEnabler.getInstance.hasUserInteracted()` value: this means that the test will successfully pass only when the creative is ad served - and will fail if running while embedded in a simple html file.
2. The creative cannot send more than **2 message per second**; the messages exceeding this frequency cap **will be automatically dropped**.

Setup In DoubleClick Studio

Following the steps above will mean you are able to fully test the ad inside of the DoubleClick Studio Preview tab both on the native Studio preview page and on external sites using Studio's 'Preview On' feature.

1. Create a new creative inside DoubleClick Studio with the relevant format and upload the file to Studio.
For more details about Rich Media format types and how to build them in Studio check here:
www.google.com/support/richmedia/bin/topic.py?topic=23546&utm_source=xuvogszn1tlltov2xl

2. Upload your assets into the Files tab, then check your creatives Events and set up the relevant Exits, Counters and Timers, before switching to the Preview tab, where you should be able to see the creative working and reacting to your mouse cursor.

Additional Notes

1. The Channel Connect component is not meant to work with many of its instances, so please instantiate just one in your creative(s).
2. Channel Connect is an experimental solution, so the suggested best practice is to prepare an alternative creative in

case an emergency backup plan is needed.
For more information, definitions, samples and examples, please visit our help centre at the following link:
www.google.com/support/richmedia

Resources

- Example FLA and MXP Component (AS3 only) - [not available yet

- DoubleClick Rich Media Help Centre
- DoubleClick Rich Media Studio
- DoubleClick Rich Media Gallery
- @rmgallery - DoubleClick Rich Media Gallery Twitter Account

Contact Us

If we haven't quite covered everything in this document, please contact your local Rich Media Campaign Manager or email dclk-drmtechnical@google.com. with any follow up questions.

About DoubleClick

For advertisers and publishers who need to reach a target audience, the DoubleClick product suite is an advertising platform that maximizes revenue growth and return on advertising spend through a unique and innovative ad targeting process. The experience and innovative spirit at DoubleClick drives a constant evolution of products and solutions, ensuring the best, most effective advertising tools are always at our customers' command.

www.doubleclick.com.

www.richmediagallery.com.

DoubleClick UK : Belgrave House, 76 Buckingham Palace Road, London SW1W 9TQ : Phone: +44 (0)800 912 1344
www.doubleclick.co.uk