EvoStream Web Services
User's Guide

Table of Contents

ABOUT THIS DOCUMENT **3**

INTENT 3
AUDIENCE 3

ABOUT THE EMS WEB SERVICES **4**

WHAT ARE EMS WEB SERVICES? 4
WHY USE THE EMS WEB SERVICES? 4
HOW DOES IT WORK? 4
SYSTEM REQUIREMENTS 4
EVENT NOTIFICATION SYSTEM 5

SETUP **7**

CONFIGURING EMS EVENT NOTIFICATIONS 7
CONFIGURING THE EMS WEB SERVICE PLUGINS 8
config.ini file *8*

EMS WEB SERVICE PLUGINS **9**

STREAM AUTOROUTING 9
STREAM RECORDER 9
STREAM LOAD BALANCER 10
AMAZON S3 HDS UPLOAD 10
AMAZON S3 UPLOAD HLS CHUNK PLUGIN 10

© 2013 EvoStream, Inc. All rights reserved.

About this Document

Intent

This document provides instructions on how to use the EvoStream Media Server (EMS) Web Services. It will cover the basics of starting the web services as well as some advanced topics like modifying configuration files.

Audience

This document is written for users of the EMS Web Services. It is expected that you have a basic understanding of multimedia streaming and the technologies required to perform multimedia streaming.

© 2013 EvoStream, Inc. All rights reserved.

About the EMS Web Services

What are EMS Web Services?

EMS Web Services are a suite of PHP scripts which can be hosted by any Web Server and that leverage the EMS's Event Notification System and Runtime-APIs. The EMS Web Services provide a convenient way to extend and customize the EMS for your projects and environments.

Why use the EMS Web Services?

The EMS Web Services are tools which use of the EMS Event Notification System together with the Runtime APIs to create custom stream processing. These PHP scripts (functions that simply wrap the HTTP interface calls of the Runtime APIs) can be used in your production environment and can be modified to meet your specific needs. The source code for all web services are provided for you to use, modify and extend!

How does it work?

The EMS Web Services are web application scripts that run with the EMS' Event Notification System and use the EMS Runtime APIs to create automated stream processing. Each web service provides a unique set of features and can be combined to create aggregate effects.

System Requirements

Operating System Requirements: The EMS Web Services can either run alongside the EMS or on a completely separate computer. The EMS Web Services can run on Windows, Linux and Mac OSX.

Web Server Requirements: The EMS Web Services require a web server, such as the Apache Web Server, along with that web server's PHP and CURL modules/plugins.

© 2013 EvoStream, Inc. All rights reserved.

Event Notification System

The EMS Event Notification System provides an extremely powerful way of interacting with the EMS. At the basic level it allows you to easily understand and monitor the usage of your server. You can either poll and parse the log file, or simply subscribe to the HTTP based notifications sent out by the EMS.

Beyond monitoring and gathering metrics, you can **use the Event Notification System to create custom stream processing.** If you want to automatically create HLS or HDS streams out of new inbound streams, simply call createHLSStream in response to each "new inbound stream" event. If you want to close inbound streams when the associated outbound stream is lost, call shutdownStream when you receive a "outbound stream closed" event.

List of Events used by the Web Services

The following events are generated by the EMS:

- **inStreamCreated** – A new inbound stream has been created
- **outStreamCreated** – A new outbound stream has been created
- **streamCreated** – A new stream has been created
- **inStreamCodecsUpdates** – A new inbound stream has been identified with a specific codec
- **outStreamCodecsUpdated** – A new outbound stream has been identified with a specific codec
- **streamCodecsUpdated** – A new stream has been identified with a specific codec
- **inStreamClosed** – An inbound stream has been closed
- **outStreamClosed** – An outbound stream has been closed
- **streamClosed** – Some stream has been closed
- **cliRequest** – The EMS has received a Runtime API command
- **cliResponse** – The response generated by the EMS for the last Runtime API command
- **applicationStart** – This event is created right after the application is initialized.
- **applicationStop** – This event is created as soon as the application has been terminated.
- **carrierCreated** – Some IO handler, such as a TCP socket, has been created
- **carrierClosed** – Some IO handler, such as a UDP socket, has been closed
- **serverStarted** – The server has started
- **serverStopped** – The server is just about to stop
- **protocolRegisteredToApp** – A connection has been fully established
- **protocolUnregisteredFromApp** – A connection has been disconnected
- **hlsChildPlaylistUpdated** – Stream specific HLS playlist has been modified
- **hlsMasterPlaylistUpdated** – HLS group playlist has been modified
- **hlsChunkCreated** – A new HLS segment was opened on disk

© 2013 EvoStream, Inc. All rights reserved.

- **hdsChildPlaylistUpdated** – Stream specific HDS manifest has been modified
- **hdsMasterPlaylistUpdated** – HDS group manifest has been modified
- **hdsChunkCreated** – A new HDS segment file has been opened
- **timerCreated** – A new timer has been created via the createTimer API command
- **timerTriggered** – The requested timer event

© 2013 EvoStream, Inc. All rights reserved.

Setup

Configuring EMS Event Notifications

EMS Event Notifications must be configured to communicate with the EMS Web Services. The EMS configuration file must be modified as follows. Please view the EMS User's Guide for more detailed information on configuring Event Notifications.

```
eventLogger=
      {
            sinks=
            {
                  {
                        type="RPC",
                        url="http://192.168.2.39/evowebservices/evowebservices.php",
                        serializerType="JSON",
                        -- serializerType="XML"
                        -- serializerType="XMLRPC"
                        enabledEvents=
                        {
                              "inStreamCreated",
                              "outStreamCreated",
                              "streamCreated",
                              "inStreamCodecsUpdated",
                              "outStreamCodecsUpdated",
                              "streamCodecsUpdated",
                              "inStreamClosed",
                              "outStreamClosed",
                              "streamClosed",
                              "cliRequest",
                              "cliResponse",
                              "applicationStart",
                              "applicationStop",
                              "carrierCreated",
                              "carrierClosed",
                              "serverStarted",
                              "serverStopping",
                              "protocolRegisteredToApp",
                              "protocolUnregisteredFromApp",
                              "processStarted",
                              "processStopped"
                        },
                  },
            },
```

© 2013 EvoStream, Inc. All rights reserved.

The **enabledEvents** parameter is optional and allows you to specify only the events which you wish to receive. **If the enabledEvents section is not specified, all events will be generated.** The list of all possible events can be found above, and more detail on each event can be found in the EMS API Definition document.

The EMS Web Services rely on EMS Events being generated as follows:

1) **RPC** – Remote Procedure Calls. Event details are transmitted to a remote host via HTTP POST. The EMS will ignore any response from the remote host.

RPC sink configuration:

```
type="RPC",
url="http://localhost/evowebservices/evowebservices.php",
serializerType="JSON"
```

The url field specifies the destination which will be accepting the HTTP POST event notifications. In this example, the web services are installed on the same computer that the EMS is running on ("localhost"). This value can be changed to the IP address of the computer running the web services.

2) **"JSON"**

Format of JSON POST:

```
{"payload":{"creationTimestamp":1349335053486.4370,"name":"","query
Timestamp":1349335053487.4370,"type":"NR","uniqueId":1,"upTime":1.0
000},"type":"streamCreated"}
```

Configuring the EMS Web Service Plugins

Every EMS Web Service is contained within a "plugin". Each plugin can be enabled or disabled, effectively turning on and off each web service. The EMS Web Service system allows many Web Service Plugins to be active at one time, allowing users to create an entire suite of features.

config.ini file

The EMS Web Services has one primary configuration file, which is located at **{DocumentRoot}\evowebservices\config\config.ini.**

Enable each web service plugin by going to the **{DocumentRoot}\evowebservices\config\config.ini.** Turn on a plugin by changing the disabled to **enabled**. **By default all plugins are disabled.**

© 2013 EvoStream, Inc. All rights reserved.

Plugin configuration:

```
[Plugins]
StreamLoadBalancer = disabled
StreamAutoRouter = disabled
StreamRecorder = disabled
AmazonHDSUpload = disabled
AmazonHLSUpload = disabled
```

EMS Web Service Plugins

Stream AutoRouting

This web service automatically forwards a stream to another EMS via RTMP. When a new stream is brought into the EMS, either by issuing a pullStream or by pushing a stream into the EMS, this web service issues an API command to automatically forward that stream to another EMS.

The AutoRouting web service has two configuration values which can be set in the config.ini file.

1) Token. If the token is defined, and not empty, the AutoRouting web service will only forward streams that have the token as part of their localStreamName.
2) Destination_uri. The address of the computer you wish to forward the streams to.

```
[StreamAutoRouter]
token = forwardthese
destination_uri = 192.168.2.58
```

Stream Recorder

This web service tells the EMS to automatically records streams.

When a new stream is created, this web service issues an API command to automatically record the stream. The user can also set how long the stream would be recorded (in seconds).

The destination for the recorded file must be set in the config.ini file.

```
[StreamRecorder]
file_location = C:\xampp\htdocs\evowebservices\media\
period_time = 60;
```

The maximum period_time is 86399 (24 hours).

© 2013 EvoStream, Inc. All rights reserved.

Stream Load Balancer

The Load Balancer web service ensures that a group of EMS instances maintain the same collection of inbound (source) streams.

When a new stream is created on one EMS, this web service will tell all the other EMS instances to also to and get that stream.

The list of EMS instances the Load Balancer will maintain is defined in the config.ini file:

```
[StreamLoadBalancer]
destination_uri[] = 192.168.2.39
destination_uri[] = 192.168.2.38
destination_uri[] = 192.168.2.58
```

Amazon S3 HDS Upload

This web service automatically uploads an HDS stream to an Amazon S3 storage instance.

As the EMS writes HDS chunks to disk, this web service uploads those file chunks to your Amazon S3 instance.

Your Amazon S3 access and secret key must be set in the config.ini file.

```
[AmazonHDSUpload]
aws_access_key = ''
aws_secret_key = ''
default_bucket = 'ems-files'
bootstrap = 'bootstrap'
```

Bootstrap is the bootstrap file which should be included with the HDS chunks. If you are unsure what this should be, it should be left as 'bootstrap'.

Amazon S3 Upload HLS Chunk Plugin

This web service automatically uploads an HLS stream to an Amazon S3 storage instance.

As the EMS writes HLS chunks to disk, this web service uploads those file chunks to your Amazon S3 instance.

Your Amazon S3 access and secret key must be set in the config.ini file.

```
[AmazonHLSUpload]
aws_access_key = ''
aws_secret_key = ''
default_bucket = 'ems-files'
```

© 2013 EvoStream, Inc. All rights reserved.