

Media Server XML Interface Documentation

\$Date: 2007-06-15 15:59:21 -0400 (Fri, 15 Jun 2007) \$ for MediaServer V03

1.0 Scope of Document

This document describes the external communication protocol for the SMS3 Media Server.

2.0 Intended Reader

This document is intended for the engineer who will write a driver for his control system to communicate with the Media Server. It is assumed that he will possess general computer programming knowledge as well as an understanding of RS-232 and/or Ethernet protocols.

3.0 Overview

The Media Server uses an XML based protocol to transmit and receive information from external controllers. This protocol is made available via RS-232 and Ethernet TCP/IP interfaces. Both the RS-232 and Ethernet interfaces are standard on the Media Server and require no additional hardware or software, however your control system may require options of RS-232 or Ethernet ports depending on the control system and choice of RS-232 or Ethernet control method used.

A limited set of commands are also available via a URL encoding method. This URL encoding method allows a standard web browser's bookmarks to send commands to the Media Server.

4.0 Media Server XML

When receiving XML commands the command is buffered until an *End of Message* (EOM) is received. When the EOM is received the entire XML command is executed.

4.1 EOM

So that we know when a message is finished being transmitted or received we send and expect an end of message marker.

The XML over RS-232 uses the “@” sign as the EOM command. It is important that the “@” sign be on its own line after any XML command's closing tag. Since the @ sign is on its own line it is safe to use the @ sign within the XML text strings sent and received from the Media Server.

The TCP interface uses either the @ sign on port 9001, or a 0x17 is used for EOM on port 9000. The streams column indicates which streams are reported and can be affected. The <stream> parameter is used to specify a stream on the ports which say 'All', and ignored on the ports which specify a stream index.

<i>Port</i>	<i>EOM</i>	<i>Streams</i>	<i>Settings</i>
RS-232	@	All	57600,8,N,1

Port	EOM	Streams	Settings
TCP port 9000	0x17	All	N/A
TCP port 9001	@	All	N/A
TCP port 9011	0x17	1	N/A
TCP port 9012	0x17	2	N/A
TCP port 9013	0x17	3	N/A
TCP port 9021	@	1	N/A
TCP port 9022	@	2	N/A
TCP port 9023	@	3	N/A

4.2 Parameters

Parameters are specified in XML as follows: `<Parameter name>value</Parameter name>`

For example, to specify stream 2: `<Stream>2</Stream>`

Parameters' values are parsed starting with the first character after the closing ">" of the parameter open-tag and ending at the last character before the "<" of the close-tag; whitespace is not collapsed. Whitespace-characters that appear inside a parameter will be taken literally and as part of the parameter's value. As such, no whitespace should appear inside a parameter unless it is to be taken as *part of* the value. In particular, note that numeric values must be free of whitespace to be considered valid.

4.3 Character Encoding UTF-8

All characters are encoded using the UTF-8 standard. This allows the use of characters beyond the standard English alphabet. This only applies to data since all commands and parameters are within 7 bit ASCII encoding. If your display device cannot display UTF-8 then you will have to convert or ignore these extended chars.

5.0 XML Transactions

There are 3 types of transactions which can occur over the XML interface:

5.1 Transaction Type - Status update

A status update is automatically sent from the Media Server whenever anything in the system changes. Note that this commonly updates even if none of the data sent has changed (it updates whenever any change occurs anywhere in the system).

It is a one way event, from the Media Server. No response is expected or required. The message is wrapped in a <MediaServerStatus> tag, with each stream wrapped in an <Info> tag.

A typical status update from a port with all streams looks like this:

Sample XML Status Update

```
<?xml version="1.0"?>
<MediaServerStatus>
  <Info>
    <Album>ShockDenialAngerAcceptance</Album>
    <AlbumCover>/music/Rick
Springfield/ShockDenialAngerAcceptance/cdcover.jpg</AlbumCover>
    <Artist>Rick Springfield</Artist>
    <Genre>Rock & Roll</Genre>
    <IDSong>36</IDSong>
    <IDTheme>3</IDTheme>
    <RunningTime>207</RunningTime>
    <State>Playing</State>
    <Stream>1</Stream>
    <Theme>New Songs</Theme>
    <Title>Eden</Title>
    <IDAlbum>3</IDAlbum>
  </Info>
  <Info>
    <Album>Breakaway</Album>
    <AlbumCover></AlbumCover>
    <Artist>Kelly Clarkson</Artist>
    <Genre>General Pop</Genre>
    <IDSong>23</IDSong>
    <IDTheme>2</IDTheme>
    <RunningTime>236</RunningTime>
    <State>Playing</State>
    <Stream>2</Stream>
    <Theme>All Songs</Theme>
    <Title>Hear Me</Title>
    <IDAlbum>2</IDAlbum>
  </Info>
  <Info>
    <Album>Rock On 1976</Album>
    <AlbumCover>/music/Various Artists/Rock On 1976/cdcover1.jpg</AlbumCover>
    <Artist>Bellamy Brothers</Artist>
    <Genre>General Rock</Genre>
    <IDSong>7</IDSong>
    <IDTheme>2</IDTheme>
    <RunningTime>171</RunningTime>
    <State>Playing</State>
    <Stream>3</Stream>
    <Theme>All Songs</Theme>
    <Title>Let Your Love Flow</Title>
```

```

    <IDAlbum>1</IDAlbum>
  </Info>
</MediaServerStatus>
EOM

```

Status Update Fields	
Tag	Description
MediaSeverStatus	Master tag around entire mesasge
Info	wrapper around each stream's status (one of these per stream in status packet)
Stream	Index of stream this Info message describes (1-3 on Media Server)
State	Current state of this stream: Playing, Stopped, Paused, Disconnected
Title	Title of currently playing song
Artist	Artist name of currently playing song
Album	Album name of currently playing song
Genre	Genre name of currently playing song
Theme	Name of currently selected Theme
IDTheme	Database Index of currently playing theme
IDSong	Database index of currently playing song
IDAlbum	Database index of album of currently playing song
AlbumCover	path to cover graphic (suitable for

Status Update Fields	
Tag	Description
	fetching via ftp). If blank then we don't have one for this album.
RunningTime	Length of song, in seconds

5.2 Transaction Type - Commands

Commands are one-shot actions which are sent to the Media Server. It is a one way event and no response will be given (unless there is an error in the command, in which case an error event will be sent back). Note that error handling is currently not very robust (many errors will just get silently ignored).

Commands are used to instruct the Media Server to do something, like skip a song, or select a new theme.

Commands are specified with a <Command> tag, followed by 0 or more parameter tags.

Sample Command - stream 1, nextsong

```
<?xml version="1.0"?>
<Event>
  <Command>NextSong</Command>
  <Stream>1</Stream>
</Event>
EOM
```

Note: Due to a name change “Themes” are sometimes referred to as “Moods” in command names and parameters. They both refer to the same thing, but some commands expect mood while others expect Theme, so be sure to adhere to the documentation on which to use. In a future release we plan to homogenize the usage to Theme, but will probably allow references to mood for backward compatibility.

5.2.1 URL Encoded Commands

It is possible to send commands using a web browser by encoding the command into the URL. Some (but not all) command transactions can be encoded in a URL. Appendix A indicates which commands can be URL encoded. Status and Query transactions are not available via this URL encoded method.

To use a URL encoded command simply enter it in a web browser's address field just like any other web address. The Media Server will execute the command and return what looks like an empty page.

Sample URL Encoded Command - stream 1, nextsong

`http://<IP>/outpers.pl?COMMAND=nextsong%20STREAM=1`

**<IP> is the IP address of the media server we wish to control*

Any command listed in *Appendix A: Media Server Commands* can be formatted as a URL string.

http://<IP>/outpers.pl

This is the base URL for the page that will execute our command. Replace **<IP>** with the IP address of the Media Server.

?COMMAND=cmd

replace **cmd** with commands from Appendix A:

%20

This is an escaped space. Space is used as a delimiter. It is escaped so that it works as a URL.

PARAMETER=value

Any parameters you wish to pass to the command are entered in this format. In the above example the parameter STREAM is set to the value 1. If multiple parameters are used a space (%20) is used as a delimiter.

Using URL encoding in a browser has some limits. For one, the resulting web page will appear empty to the user. A command line (DOS) web page fetching utility can be used to execute URLs without opening a window. WGET is a popular utility that can be easily found on the web. This is useful for desktop or toolbar icons that silently send commands to the Media Server. To construct such a desktop icon will require the creation of a DOS batch file that executes the wget command to fetch a URL encoded command. The batch file can then be attached to a desktop icon.

[See Appendix A:Media Server Commands](#)

5.3 Transaction Type - Queries

Queries are sent to the Media Server, and a response is sent back. It is a two-way event.

Queries never change the state of the system, just allow information to be read from the system.

Queries support custom tags. These custom tags are echoed back to the caller when the query is returned. Typically these are used to execute a callback function, or to set an ID for the query. Note that the returned tags will be lower cased. This may be changed in a future release, so the safest thing to do is to use all lower case tags.

Queries are requested by sending the "sendquery" command. The type parameter specifies what type of query to execute.

Sample query fetching titles starting with S, limited to 5 results.

Notice the "anynameworthy" parameters. These are custom tags.

```
<?xml version="1.0"?>
<Event>
```

```
<Command>sendquery</Command>
<Type>searchtitles</Type>
<letters>S</letters>
<quantity>5</quantity>
<anynameyoulike>any value you like</anynameyoulike>
</Event>
EOM
```

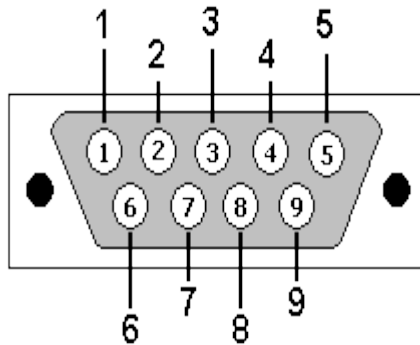
Sample result

```
<?xml version="1.0"?>
<QueryData>
  <letters>S</letters>
  <quantity>5</quantity>
  <anynameyoulike>any value you like</anynameyoulike>
  <type>searchtitles</type>
  <Item0>
    <IdSong>845</IdSong>
    <Title>Sweet Layabout</Title>
    <Artist>Soup Dragons</Artist>
    <Album>Hotwired</Album>
    <Genre>General Rock</Genre>
  </Item0>
  <Item1>
    <IdSong>6</IdSong>
    <Title>SupeuTrouper</Title>
    <Artist>Abba</Artist>
    <Album>Gold Greatest Hits</Album>
    <Genre>Unknown</Genre>
  </Item1>
  <Item2>
    <IdSong>1</IdSong>
    <Title>S. O. S.</Title>
    <Artist>Abba</Artist>
    <Album>Gold Greatest Hits</Album>
    <Genre>Unknown</Genre>
  </Item2>
  <Item3>
    <IdSong>27</IdSong>
    <Title>Satori</Title>
    <Artist>ABC</Artist>
    <Album>Abracadabra</Album>
    <Genre>Unknown</Genre>
  </Item3>
  <Item4>
    <IdSong>25</IdSong>
    <Title>Say it</Title>
    <Artist>ABC</Artist>
    <Album>Abracadabra</Album>
    <Genre>Unknown</Genre>
  </Item4>
</QueryData>
EOM
```

[See Appendix B: Media Server Queries](#)

6.0 RS-232 Hardware

If connecting the Media Server to a PC via RS-232, a “Null Modem” cable is required Radio Shack Catalog #**26-264** (null adapter), and #**950-0225** (6' cable). Other devices may require a custom cable.



<i>Pin No.</i>	<i>Name</i>	<i>Notes/Description</i>
1	DCD	Data Carrier Detect
2	RD	Receive Data (a.k.a RxD, Rx)
3	TD	Transmit Data (a.k.a TxD, Tx)
4	DTR	Data Terminal Ready
5	SGND	Ground
6	DSR	Data Set Ready
7	RTS	Request To Send
8	CTS	Clear To Send
9	RI	Ring Indicator