TCP Protocol for Resolve Transport Control

46

TCP Protocol for Resolve Transport Control

This chapter describes how to create third party utilities that use Transport Control with DaVinci Resolve.

About the TCP Protocol Version 1.2

This protocol defines the communication standard between third party applications (“Client”) and DaVinci Resolve (“Server”) using the TCP protocol.

Port number 9060 will be used by the server. SSL will not be used in this protocol. Communication takes the form of request-response messages, where the Client initiates a command, and the Server responds appropriately.

Data Types

The following data types are used in this protocol:

- float (f): A 4-byte IEEE 754 single precision float
- int (i): A 4-bytes signed int
- unsigned char (uc): A 1-byte unsigned char (0-255)
- string (s): A UTF-8 encoded string. No terminator is specified. The string is a composite type, transmitted as a single int (i) specifying the number of characters in the string (N), followed by N unsigned chars (uc) containing the letters of the string.

NOTE: The bytes of the float and int types are transmitted in little endian order.

Command Format

Commands are transmitted as a single string (using characters a-z (0x61 – 0x7A) only), followed by any additional payload required by the command in the definition.

Response Format

The response to any command is composed of a status byte (unsigned char), followed by any additional payload required by the response.

Communication Delays

Once the first byte of the command string is sent, the rest of the command string and the payload data must follow without delay. At the end of COMMAND, the server must respond immediately. If there is a delay of more than 5 seconds during this process, the party waiting for data may drop the connection assuming that the peer has become unresponsive.

There is currently no limit on the delay between two consecutive commands.

NOTE: Alternatively, a maximum allowable delay may be defined, in which case, the client may issue periodic ‘connect’ commands to keep the connection alive.

Status Response Values

The meaning of the status values are as follows:

- 0x00: Command was executed successfully. Any additional payload is sent as expected.
- 0xFF: Command could not be executed successfully. No additional payload will follow.

TCP Protocol Stream

The following commands can be sent over the protocol stream.

connect

The client initiates the stream by sending a connect command string. There is no payload. The server responds with a status value of 0x00.

goto

The client sends a goto command string followed by four unsigned chars representing the hour, minute, second, and frame of the timecode.

The server responds with an appropriate status byte based on the execution of the command.

play

The client sends a play command string followed by a floating point value. Play in real-time is 1.0, stop is 0.0, reverse is -1.0, 2x is 2.0, etc.

The server responds with an appropriate status byte based on the execution of the command.

gettc

The client sends a gettc command string.

The server responds with an appropriate status byte (status byte may be 0xFF if no timeline exists, for instance). If the status byte is 0x00, it is followed by four unsigned chars representing the hour, minute, second, and frame of the timecode.

getframerate

The client sends a getframerate command string.

The server responds with an appropriate status byte. If the status byte is 0x00, it is followed by a floating point value for the frame rate.