TechNote TN78:   Sooloos Control Protocol v1.10

Protocol lines end in \n and begin with either $, *, or !

Unsolicited events can occur at any time, even before a response is read for a command.

Three primitive types are used in the protocol commands and events:

[int]   A valid integer in string form.

[string] A string value. Strings do not / must not contain \n. All strings can be contained within double quotes (e.g. "value"). Empty strings are represented by "". Double quotes within a string are prefixed with a backslash (\"). Blackslashes (\) are encoded as double backslash (\\).

[sid]   Sooloos ID. All albums and songs are represented by a Sooloos ID string. All ID strings are unique strings which represent that particular album or song. All commands and events dealing with albums and songs use Sooloos ID strings. These strings are contained within double quotes (e.g. "id").

The general format of this document for each command, event, or response is as follows:

```
<command name>
     <documentation>
<syntax>
    [<parameter descriptions>]
[return format]
    [return format parameter description]
```

If a response contains N number of fields, you should not assume it will always contain N. In future revisions of the protocol, it may contain more than N, and to be future-proof, it's a good idea to ignore the fields after the N in this document, instead of breaking.

_____

These commands are sent to the Sooloos System:
_____

PING
        Ping the interface to check that it is alive.
$PNG

RATE LIMIT
        Limit the rate that sooloos sends data over the TCP socket connection.

        This should only be used as a workaround for clients that, due to limitations in their programming environments, cannot use TCP flow control to limit the rate of data.
        By allowing small bursts of data, this should not have a significant impact on small amounts of data.

$RAT burst cool_off
        burst   [int] The maximum number of bytes that can be sent at once
        cool_off[int] The number of milliseconds to wait after sending [burst] bytes

ZONE RESPONSE TYPE

This command tells the protocol to send zone names in the responses as keys instead of as their names.

$ZRT type

    type   [int] 0 = send names, 1 = send keys

PLAY QUEUE ADD SONG

This command adds an individual song to the play queue optionally specifying when the song should play. The default is to queue songs for play.

$PAS zone song play

    zone   [string] Zone name
    song   [sid] Sooloos ID of the song
    play   [int] 0 = queue, 1 = play next, 2 = play now

PLAY QUEUE ADD ALBUM

This command adds an entire album to the play queue optionally specifying when the songs should play. The default is to queue the songs for play. The command specifies if all songs in the album should be played or if only album picks (favorites) should be played.

$PAA zone album what play

    zone   [string] Zone name
    album [sid] Sooloos ID of the album
    what   [int] 0 = all songs, 1 = album picks
    play   [int] 0 = queue, 1 = play next, 2 = play now

PLAY QUEUE LOOP

This command modifies the current loop behavior of the play queue. The loop can be turned on/off, or toggled from its current value.

$PLO zone what

    zone   [string] Zone name
    what   [int] 0 = off, 1 = on, 2 = toggle

PLAY QUEUE SHUFFLE

This command modifies the current shuffle behavior of the play queue. The shuffle can be turned on/off, or toggled from its current value.

$PSH zone what

    zone   [string] Zone name
    what   [int] 0 = off, 1 = on, 2 = toggle

PLAY QUEUE SWIM DISABLE

This command disables the swim behavior of the play queue. This command has no effect if swim is currently not enabled on the play queue.

$PSD zone

    zone   [string] Zone name

PLAY QUEUE EMPTY

This command empties the play queue for the zone.

$PQE zone

    zone   [string] Zone name

TRANSPORT PLAY OR PAUSE

This command instructs the server to begin playing the zone or pause the zone if it happens to already be playing.

$TPP zone

    zone   [string] Zone name

TRANSPORT PLAY

This command instructs the server to begin playing the zone.

$TPL zone

    zone   [string] Zone name

TRANSPORT PAUSE

This command instructs the server to pause or unpause the zone or simply toggle its current value.

$TPA zone what

    zone   [string] Zone name
    what   [int] 0 = off, 1 = on, 2 = toggle

TRANSPORT STOP

This command instructs the server to stop the zone.

$TST zone

    zone   [string] Zone name

TRANSPORT ADVANCE

This command instructs the server to advance the zone transport eitherforward or backward one song.

$TAD zone what

    zone   [string] Zone name
    what   [int] 0 = next, 1 = previous

TRANSPORT SEEK

This command instructs the server to seek to a position in the playing song. By default, the song position is an absolute offset from the start of the song. If the optional relative parameter is specified, the position is a positive or negative number of seconds to offset from the current song position.

$TSK zone seconds rel

    zone   [string] Zone name to seek
    seconds [int] Number of seconds offset from the song start
    rel   [int] 0 = absolute from start, 1 = relative from current position

VOLUME UP

This command instructs the server to increase the volume of the zone by 1 unit.

$VUP zone

    zone   [string] Zone name

VOLUME DOWN

This command instructs the server to decrease the volume of the zone by 1 unit.

$VDN zone

    zone   [string] Zone name

VOLUME MUTE

This command instructs the server to toggle the mute state of the zone.

$VMU zone

    zone   [string] Zone name

COUNT ZONES (Returns count)

This command requests a count of the system audio zones.

$CZN

COUNT ARTISTS (Returns count)

This command requests a count of the number of artists.

$CAR

COUNT ALBUMS FOR ARTIST (Returns count)

This command requests a count of the number of albums for a given artist in the system.

$CAL artist

    artist   [string] Arist name

COUNT SONGS FOR ALBUM (Returns count)

This command requests a count of the number of songs for a given album in the system.

$CSN album

    album   [sid] Sooloos ID of the album

DUMP PLAYING STATE

This command instructs the server to dump the state of loop, shuffle, swim, playing, paused, and stopped for this zone by sending the appropriate '!' events.

$DPT sends all of the following: !PSH !PLO !PSW
$DPT sends one of the following: !TPL !TPA !TST

$DPT

    zone   [string] Zone name

DUMP ZONES (Returns data)

This command instructs the server to dump a listing of all available zones. Each data line contains the zone name which is used as parameters in many other commands.

$DZN

    returns zones in the form: zone
    zone   [string] Zone name
    key   [string] Zone key

UNIVERSE ACTION

This command executes an action retrieved via the $DUN commands.

$UNA action [zone]

    action   [string] Action command text, see $DUN
    zone   [string] Zone name, needed only if action requires it

DUMP UNIVERSE (Returns data)
>This command instructs the server to dump a listing of a refinement of the universe. Note
>that clients will probably use either $DUR or $DUN but not both.

$DUN refinement n p
>refinement [string] a refinement from a previous $DUN/DUR command or ""
>n   [int] Number of items per page to dump
>p   [int] Page number to dump

returns N nodes of data of the form: display refinement count ACTIONS
>display   [string] Item's display text
>refinement [string] Refinement string passed back to $DUN
>count   [int] The number of actions to follow
>ACTIONS [special] Actions are encoded one per line in the form:
>>zone confirm conf_text reload name command
>>zone   [char] 'M'usic 'V'ideo 'N'one
>>confirm   [char] 'C'onfirm 'N'one
>>conf_text   [string] Confirmation question to ask. Any $$ found in the string should
>>>be replaced by the node display text.
>>reload   [char] 'R'eload 'N'one
>>name   [string] Action display name
>>command   [string] Action command to pass to $UNA

DUMP ARTISTS (Returns data)
>This command instructs the server to dump a listing of all artists currently in the system.

$DAR n p
>n   [int] Number of items per page to dump
>p   [int] Page number to dump

returns N artists of the form: name album_cnt
>name   [string] Artist name
>album_cnt [int] Number of albums by the artist

DUMP ARTIST ALBUMS (Returns data)
>This command instructs the server to dump a listing of all albums for a given artist in the
>system.

$DAL artist n p
>artist   [string] Artist name to dump
>n   [int] Number of items per page to dump
>p   [int] Page number to dump

returns N albums of the form: album_sid album_name song_cnt
>album_sid   [sid] Sooloos ID of the album
>album_name [string] Name of the album
>song_cnt   [int] Number of song in the album

DUMP ALBUM SONGS (Returns data)
>This command instructs the server to dump a listing of all songs for a given album in the
>system.

$DSN album [n p]
>album   [sid] Sooloos ID of the album
>n   [int] Number of items per page to dump
>p   [int] Page number to dump

returns N songs of the form: track_number song_sid song_name length flags
>track_number   [int] Track number (1-99)
>song_sid   [sid] Sooloos ID of the song

<pre>
        song_name   [string] Name of the song
        length   [int] Length of the song in seconds
        flags   [int] The follwing flags are bitwise OR'd together:
                         0x01 = song pick
                         0x02 = never play
                         0x04 = never been played
</pre>

DUMP ALBUM COVER URL (Returns data)

This command instructs the server to dump a URL representing the cover art for a given album in the system. If no lines of data are returned there is no cover available. The URL is a standard http:// prefixed URL.

$DCV album
<pre>
        album   [sid] Sooloos ID of the album
</pre>
returns 0 or 1 URL of the form: url
<pre>
        url   [string] URL of the album cover
</pre>

DUMP PLAYING SONG (Returns data)

This command instructs the server to dump a single data line containing the currently playing song within the play queue.

$DPS zone
<pre>
        zone   [string] Zone name
</pre>
returns 1 song of the form: song_sid song_name album_sid album_name artist_name length pos playing
<pre>
        song_sid   [sid] Sooloos ID of the song
        song_name   [string] Name of the song
        album_sid   [sid] Sooloos ID of the album
        album_name   [string] Name of the album
        artist_name   [string] Name of the artist
        length   [int] Length of the song in seconds
        pos   [int] Current song position in seconds
        playing   [int] 0 = stopped, 1 = playing, 2 = paused
        url   [string] URL of the album cover
</pre>

Lightweight universe browsing

All light universe commands begin with $LU. Commands not returning data return *AOK for success. Other responses are described below.

INITIALIZE UNIVERSE BROWSING SESSION

This command initializes the stateful universe browser with an arbitrary session id supplied by the client. Different session ids may be used to maintain multiple browsing sessions concurrently on the same TCP connection

$LUI session truncate [enable_search]
<pre>
        session   [string] arbitrary session id to identify this browsing session
        truncate [int] number of characters to truncate display strings to
                  0 means perform no truncation. If strings are truncated, then'...' is appended such
                  that the truncated string, including the '...', fits within the specified limit.
                      Note that the truncation limit reflects UTF-8 characters and not bytes as
                      characters in UTF-8 may consume multiple bytes.
</pre>

enable_search
    [int] (optional) 0 means disable search, 1 means enable search
        if search is enabled then $LUN will return both search nodes and refinement nodes.

returns the root of the universe as an *LUL response

LOAD UNIVERSE SEARCH
    This command loads a universe search based on an index into the current universe list
$LUS session level index query
    session [string] arbitrary session id to identify this browsing session
    level   [int] The level in the universe to fetch nodes for. This value cannot exceed the
            current level of the universe as reported by the most recent *LUL response.
    index   [int] index into the current universe list from which to load the search
    query   [string] String representing the search terms
returns the newly-loaded level of the universe as an *LUL response

LOAD UNIVERSE REFINEMENT
    This command loads a universe refinement based on an index into the current universe list
$LUR session level index
    session [string] arbitrary session id to identify this browsing session
    level   [int] The level in the universe to fetch nodes for. This value cannot exceed the
            current level of the universe as reported by the most recent *LUL response.
    index   [int] index into the current universe list from which to load the refinement
returns the newly-loaded level of the universe as an *LUL response

MOVE BACK TO PREVIOUS UNIVERSE LEVEL
    This command moves back one level in the universe. Attempts to move back beyond the root
    of the universe reload the root level.
$LUB session
    session [string] arbitrary session id to identify this browsing session
returns the newly-loaded level of the universe as an *LUL response

LOAD UNIVERSE REFINEMENT NODES
    This command returns one or more nodes from a universe list.
$LUN session level offset count
    session [string] arbitrary session id to identify this browsing session
    level   [int] The level in the universe to fetch nodes for. This value cannot exceed the
            current level of the universe as reported by the most recent *LUL response.
    offset   [int] Offset into list to dump
    count   [int] Count of nodes to dump
returns COUNT nodes of data of the form: display isrefinement actions
    display [string] Display string
    node_type   [int] 0 = nothing
                1 = node can be fetched as a refinement using $LUR
                2 = node can be fetched as a search using $LUS
            search nodes will only be returned if search was
            enabled by the $LUI command for this session
    actions [int] the number of actions associated with this node

LOAD UNIVERSE ACTIONS

This command returns one or more actions from a universe node

$LUA session level index offset count

session [string] arbitrary session id to identify this browsing session

level   [int] The level in the universe to fetch actions for. This value cannot exceed the current level of the universe as reported by the most recent *LUL response.

index   [int] index into the current universe list

offset   [int] offset into the list of actions for this node

count   [int] count of actions to return

returns COUNT actions in the form: zone confirm conf_text reload display

zone   [char] 'M'usic 'V'ideo 'N'one

confirm [char] 'C'onfirm 'N'one

conf_text [string] Confirmation question to ask. Any $$ found in the string should be replaced by the node display text.

reload   [char] 'R'eload 'N'one

display [string] Action display name

EXECUTE UNIVERSE ACTION

This command executes a universe action

$LUE session level index action [zone]

session [string] arbitrary session id to identify this browsing session

level   [int] The level in the universe to fetch actions for. This value cannot exceed the current level of the universe as reported by the most recent *LUL response.

index   [int] index into the current universe list

action   [int] index into actions list for the node at index

zone   [string] Zone name, needed only if action requires it

returns *AOK to indicate success or *ERR to indicate failure.

UNIVERSE LEVEL QUERY

This command requests the active level of the current universe list.

$LUQ session

session [string] arbitrary session id to identify this browsing session

returns the active level of the universe as an *LUL response

END UNIVERSE BROWSING SESSION

This command ends a universe browsing session, discarding all state associated with it.

$LUX session

_____

These are the responses from the Sooloos System:
_____

ERROR

This generic response can be sent for any command which does not complete successfully for whatever reason. The error will always be the first and only response for any command which does not complete.

*ERR text

text   [string] the error message text

Commands not returning data or count:

SUCCESS
This generic response is sent for any command which completes successfully and does not
return a count ($Cxx) or return data ($Dxx).
Commands which return a count or data do not send an *AOK response.
*AOK

PING
This response is sent after receipt of the $PNG command to the client that sent the ping.
*PNG

Commands returning count ($Cxx):

COUNT RETURNED
This response is sent to indicate the number of items in whatever set is requested by the
various $Cxx commands.
*ACN count
    count   [int] The count of items in the set

Commands returning data ($Dxx):

DATA FOLLOWING
This response is sent to indicate that the data request was successful and data (*DAT)
responses will follow, followed by a data stop (*DAS) response.
*DAF

DATA
This response is sent once for each item of data being returned. It is optional if there are no
items to return. This response is followed by a data stop once the data dump is complete.
The data format in this response is a whitespace separated list of fields as defined by the
return value of every $Dxx command.
*DAT data
    data   [string] The return data (format is command specific)

DATA STOP
This response is sent to indicate that there are no more data (*DAT) responses and the data
dump is complete. An empty data set will respond with *DAF followed by *DAS.
*DAS

UNIVERSE LEVEL RESPONSE
This response is sent to indicate the number of nodes in a universe list
*LUL count level display
    count   [int] The count of nodes in the set
    level   [int] The 'level' of the universe that we are browsing.
       0 is the root level.
    display [string] The display string for this universe level.

_____

These unsolicited events are sent by the Sooloos System at any time.
_____

PLAY QUEUE CURRENT SONG CHANGED
>This event is sent whenever the currently playing song in the play queue changes. This event indicates clients need to update the display of the currently playing song.

!PCS zone song_sid song_name album_name artist_name song_length
zone	[string] The zone name which changed
song_sid	[sid] Sooloos ID of the song
song_name	[string] Name of the song
album_name	[string] Name of the album
artist_name	[string] Name of the artist
length	[int] Length of the song in seconds
url	[string] URL of the album cover

PLAY QUEUE TIME REMAINING
>This event is sent every second for every actively playing zone so that clients can keep play queue time remaining values in sync.

!PTR zone time
zone	[string] The zone name which changed
time	[int] The number of seconds remaining in the play queue

PLAY QUEUE LOOP CHANGED
>This event is sent whenever any client modifies the loop setting. This command is echoed back to the client which requested the loop setting be changed.

!PLO zone what
zone	[string] The zone name which changed
what	[int] 0 = off, 1 = on

PLAY QUEUE SHUFFLE CHANGED
>This event is sent whenever any client modifies the shuffle setting. This command is echoed back to the client which requested the shuffle setting be changed.

!PSH zone what
zone	[string] The zone name which changed
what	[int] 0 = off, 1 = on

PLAY QUEUE SWIM CHANGED
>This event is sent whenever any client modifies the swim setting. This command is echoed back to the client which requested the swim setting be disabled.

!PSW zone what
zone	[string] The zone name which changed
what	[int] 0 = off, 1 = on

REFRESH ZONES
>This event is sent whenever zones are removed or added to the system and the client should re-request the zone listing.

!RZN

REFRESH COLLECTION
>This event is sent whenever a server state change has modified the music collection. All artist, album and song information should be invalidated and re-requested if necessary.

!RCO

TRANSPORT PLAYING

This event is sent whenever any client instructs a zone to begin playing. This command is echoed back to the client which requests the zone to play.

!TPL zone

zone   [string] The zone name which began playing

TRANSPORT PAUSED

This event is sent whenever any client instructs a zone to pause. This command is echoed back to the client which requests the zone to pause.

!TPA zone

zone   [string] The zone name which paused

TRANSPORT STOPPED

This event is sent whenever any client instructs a zone to stop playing. This command is echoed back to the client which requested the zone to stop.

!TST zone

zone   [string] The zone name which stopped

TRANSPORT SEEK POSITION

This event is sent every second for every actively playing zone so that clients can keep song progress displays updated.

!TSK zone seconds

zone   [string] The zone name being updated
seconds [int] The current song position in seconds

VOLUME INCREASED

This event is sent whenever any client instructs a zone to increase its volume. This command is echoed back to the client which requested the zone to increase in volume.

!VUP zone volume

zone   [string] The zone name that increased its volume
volume [string] The current volume with the format depending on volume control type of the zone.

VOLUME DECREASED

This event is sent whenever any client instructs a zone to decrease its volume. This command is echoed back to the client which requested the zone to decrease in volume.

!VDN zone volume

zone   [string] The zone name that increased its volume
volume [string] The current volume with the format depending on volume control type of the zone.

VOLUME MUTE STATE CHANGED

This event is sent whenever any client instructs a zone to change its mute state. The command is echoed back to the client which requested the change in mute state.

!VMU zone state

zone   [string] The zone name that changed its mute state
state   [string] The current mute state of the zone, either MUTED or UNMUTED