

ePLAYER1 JSON Commands

THIRD-PARTY CONTROL PROTOCOL



PRODUCT OVERVIEW

JSON (JavaScript Object Notation) is a lightweight data-interchange format that allows ePLAYER1 to communicate with third-party devices and platforms, such as EclerNet Manager. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages. Visit the official website for more information: <https://www.json.org>

VERSIONS

- V1.0 [OCT20] – Latest release
- Preliminary version

LIST OF COMPATIBLE DEVICES

- ePLAYER1

GENERAL CONSIDERATIONS

- The communication with can be established using Ethernet or WiFi and the TCP/IP transport protocol, always by means of the **2003 TCP port**.
- To check IP address, hold ENTER button for 2-3 seconds to enter admin menu.
- To let some control systems (like CRESTRON®, EXTRON®, AMX®, RTI®, VITY®, MEDIALON®, etc.) process the messages more easily, ePLAYER1 allows to the end of each message with a CR (**\n**) -line feed, character 10-.
- All commands answer {"result":true} (success) or {"result":false} (something failed)

PLAYER COMMANDS

SET PLAYER STEREO/MONO

Mono mode

```
{"jsonrpc":"2.0","method":"Player.Stereo","Stereo":false}
```

Stereo mode

```
{"jsonrpc":"2.0","method":"Player.Stereo","Stereo":true}
```

SET PLAYER FADE

No fade

```
{"jsonrpc":"2.0","method":"Player.Fade","Fade":0}
```

Cross Fade

```
{"jsonrpc":"2.0","method":"Player.Fade","Fade":1}
```

Fade

```
{"jsonrpc":"2.0","method":"Player.Fade","Fade":2}
```

SET PLAYER MODE

Player mode Sequential

```
{"jsonrpc":"2.0","method":"Player.Mode","PlayMode":0}
```

Player mode Random

```
{"jsonrpc":"2.0","method":"Player.Mode","PlayMode":1}
```

SET PLAYER REPEAT

Play all

```
{"jsonrpc":"2.0","method":"Player.Repeat","Repeat":0}
```

Play one

```
{"jsonrpc":"2.0","method":"Player.Repeat","Repeat":1}
```

Repeat all

```
{"jsonrpc":"2.0","method":"Player.Repeat","Repeat":2}
```

Repeat one

```
{"jsonrpc":"2.0","method":"Player.Repeat","Repeat":3}
```

GET SHORT PLAYER INFORMATION

```
{"jsonrpc":"2.0","method":"Player.GetStats"}
```

```
{"title":"Brian Hyland - Sealed With a  
Kiss","counter":"19:30","txtSource":"NET","status":1}
```

GET FULL PLAYER INFORMATION

```
{"jsonrpc":"2.0","method":"Player.GetStatsEx"}
```

```
{"title":"Elvis Presley - Judy","counter":"07:02","txtSource":"NET","status":1,"SourceList":  
["","MMC","USB UNAVAILABLE","DLNA","AIRPLAY","JVL  
PLAYLIST","MUSICUP"],"source":6,"preset":1,"volume":100,"txtVolume":"0dB","stereo":1,"repeat":  
2,"playmode":0,"fade":1,"bootpreset1":0,"sp":1,"bitrate":"128","duration":"--:--  
","freq":"44.1","playlist_index":" 0006 / 0056"}
```

PLAYER ADD NEXT PLAYLIST ITEM

With this function user can manage device playlist simply inserting next item before the end of current item.

set next item to "next_item.mp3"

```
{"jsonrpc": "2.0", "method": "Player.QueueNextElem", "url": "mmc://next_item.mp3"}
```

PLAYER INSERT PRORITY ITEM

With this function user can insert a priority item that will be played "over" the actual playing item. Current playing item will be fade.

set next item to "priority_item.mp3"

```
{"jsonrpc": "2.0", "method": "Player.PrioritySetElem", "url": "usb://priority_item.mp3"}
```

PLAYER PLAY

If the player is paused or stopped use this function to start current loaded item reproduction, otherwise the player is paused.

```
{"jsonrpc": "2.0", "method": "Player.Play"}
```

PLAYER STOP

```
{"jsonrpc": "2.0", "method": "Player.Stop"}
```

PLAYER NEXT

```
{"jsonrpc": "2.0", "method": "Player.Next"}
```

PLAYER PREVIOUS

```
{"jsonrpc": "2.0", "method": "Player.Prev"}
```

INCREMENT VOLUME

Increment volume just one dB

```
{"jsonrpc": "2.0", "method": "Player.Volume", "Action": "inc"}
```

DECREMENT VOLUME

Decrement volume just one dB

```
{"jsonrpc": "2.0", "method": "Player.Volume", "Action": "dec"}
```

SET VOLUME

Param volume is expressed in %. To set volume to 50% use next command.

```
{"jsonrpc": "2.0", "method": "Player.Volume", "Volume": 50}
```

OPEN PLAYLIST URL

Url param must be any valid device url.

```
{"jsonrpc": "2.0", "method": "Player.Open", "Url": "http://50.7.181.186:8060"}
```

OPEN PRESET INDEX 10

Preset param must be a valid preset index 1 to 20

```
{"jsonrpc": "2.0", "method": "Player.Open", "Preset": 10}
```

OPEN SOURCE INDEX 4 AIRPLAY (FROM AVAILABLE SOURCES LIST)

Source must be a valid player source index. Please check “Get of list available sources” to know all valid sources.

```
{"jsonrpc": "2.0", "method": "Player.Open", "Source": 4}
```

GET LIST OF AVAIABLE SOURCES

This command returns the list of available sources.

```
{"jsonrpc": "2.0", "method": "Source.GetList"}
```

```
{"SourceList": [ "", "MMC", "USB UNAVAILABLE", "DLNA", "AIRPLAY", "ROCK 80s", "DISCO 80s" ]}
```

CONFIGURATION COMMANDS

RESET DEVICE SETTINGS

Restore device factory default settings. All your settings will be lost.

```
{"jsonrpc": "2.0", "method": "Settings.Reset"}
```

RESTORE DEVICE SETTINGS FROM URL

Restore device settings to values in url file.

```
{"jsonrpc": "2.0", "method": "Settings.Restore", "url": "http://ecler.com/my\_player\_config.config"}
```

BACKUP CURRENT DEVICE CONFIGURATION

Backup device settings to url. Available configurations: user, admin or gallery.

```
{"jsonrpc": "2.0", "method": "Settings.Backup", "url": "mmc://backups/gim.config", "user": "admin"}
```

GETTING DEVICE VARIABLE VALUE

This function returns a device variable value. Please check the LUA ePLAYER1 manual in order to check all the interface.settings.variable values.

In order to retrieve preset01.settings.bname (preset name) user should send next command to the ePLAYER1:

```
{"jsonrpc": "2.0", "method": "CFG.get", "interface": "preset01", "section": "settings", "variable": "bname"}
```

```
{"value": "AFTERNOON PRESET"}
```

SETTING DEVICE VARIABLE VALUE

This function set a device variable value. Please check the LUA ePLAYER1 manual in order to check all the interface.settings.variable values.

In order to set preset01.settings.bname (preset name) user should send next command to the ePLAYER1:

```
{"jsonrpc": "2.0", "method": "CFG.set", "interface": "preset01", "section": "settings", "variable": "bname", "value": "MIDNIGHT PRESET"}
```

STORE CHANGES IN DEVICE INTERNAL MEMORY

This function stores all interface variables to the internal device memory. Should be called after set all the changes. ePLAYER1 must reload the data using [reload functions](#).

```
{"jsonrpc": "2.0", "method": "CFG.commit", "interface": "preset01"}
```

PRESET COMMANDS

Preset variables for GFG.get and CFG.set. Please check [ePLAYER1.pdf](#) Preset variables.

RELOAD PRESET

Reload indicated preset index. Index should be a valid preset index 1..20. Must be called after modifying preset variables and call commit command

```
{"jsonrpc": "2.0", "method": "Preset.Reload", "Index": 1}
```

EVENT COMMANDS

Event variables for GFG.get and CFG.set. Please check [ePLAYER1.pdf](#) Event variables.

RELOAD EVENT

Reload indicated event. Name should be: GPI1, GPI2 or SILENCE. Must be called after modifying event variables and call commit command.

```
{"jsonrpc": "2.0", "method": "Event.Reload", "Name": "GPI1"}
```

CALENDAR COMMANDS

Calendar variables for GFG.get and CFG.set. Please check [ePLAYER1.pdf](#) Preset variables

RELOAD CALENDAR

Reload indicated calendar. Calendar index should be a number 1..24. Must be called after modifying calendar variables and call commit command. Reload calendar 24 example:

```
{"jsonrpc": "2.0", "method": "Calendar.Reload", "Index": 24}
```

STORE AND FORWARD COMMANDS

SAF variables for GFG.get and CFG.set. Please check [ePLAYER1.pdf](#) SAF variables

RELOAD SAF

Reload SAF configuration. Must be called after modifying SAF variables and call commit command.

```
{"jsonrpc": "2.0", "method": "SAF.Reload"}
```

GOOGLE DRIVE COMMANDS

SAF variables for GFG.get and CFG.set. Please check [ePLAYER1.pdf](#) SAF variables

RELOAD GOOGLE DRIVE

Reload GDRIVE configuration. Must be called after modifying Google Drive variables and call commit command.

```
{"jsonrpc": "2.0", "method": "GDRIVE.Reload"}
```

AUTHENTICATE GOOGLE DRIVE

Call this command to validate Google Drive configuration with Google servers.

```
{"jsonrpc": "2.0", "method": "GDRIVE.Authenticate"}
```

SYNCHRONIZE GOOGLE DRIVE

Call this function to synchronize now Google Drive content

```
{"jsonrpc": "2.0", "method": "GDRIVE.Synchronize"}
```

CONTENT MANAGEMENT SYSTEM (CMS) COMMANDS

CMS variables for GFG.get and CFG.set. Please check [ePLAYER1.pdf](#) CMS variables.

RELOAD CMS

Reload CMS configuration. Must be called after modifying CMS variables and call commit command.

```
{"jsonrpc": "2.0", "method": "CMS.Reload"}
```

SCRIPTS COMMANDS

Script variables for GFG.get and CFG.set. Please check [ePLAYER1.pdf](#) Script variables.

RELOAD SCRIPT

Reload script configuration. Index should be script index 1 to 20. Must be called after modifying Script variables and call commit command. Reload script 7 example:

```
{"jsonrpc": "2.0", "method": "Script.Reload", "Index": 7}
```

EXECUTE SCRIPT 6

```
{"jsonrpc":"2.0","method":"Script.Command","Index":6,"Command":"Start"}
```

KILL SCRIPT 3

```
{"jsonrpc":"2.0","method":"Script.Command","Index":3,"Command":"Stop"}
```

QUERY SCRIPT 11 STATUS

```
{"jsonrpc":"2.0","method":"Script.Status","Index":11}  
{"status":"Idle"}
```

REGISTER COMMANDS

ADD REGISTER LINE

Add line to device LOG. Possible line values are: Trace, Warning, Error.

Add a warning line example:

```
{"jsonrpc":"2.0","method":"Device.Log","Severity":"Trace","Message":"This is a warning  
message"}
```

DEVICE COMMANDS

DEVICE REBOOT

```
{"jsonrpc":"2.0","method":"Device.Reboot"}
```

GET DEVICE VERSION

```
{"jsonrpc":"2.0","method":"Device.GetVersion"}  
{"version":"3.04r0"}
```

DEVICE UPDATE FIRMWARE

With this function user could update device firmware to an specific version. User must provide firmware url. Device setting will be saved.

```
{"jsonrpc":"2.0","method":"Device.Update","url":"https://www.ecler.com/new_firmware.bin"}
```

DEVICE BOOT CONFIG COMMAND

Available BootPreset1 options are: 1 – PRESET1, 2 – keep status

```
{"jsonrpc":"2.0","method":"Device.BootPreset1","BootPreset1":2}
```

DEVICE GET MAC

```
{"jsonrpc":"2.0","method":"Device.GetMac"}  
{"mac":"32 41 41 20 40 42"}
```

DEVICE GET GALLERY REGISTRATION KEY

```
{"jsonrpc": "2.0", "method": "Device.GetRegkey"}  
{"regkey": "2E1BB146B2DB2WA1"}
```

PANEL COMMANDS

PANEL SET LOCK SETTINGS

Set panel Lock to "UNLOCK ALL" "UNLOCK USER" "LOCK ALL". Set panel password to Pass.

```
{"jsonrpc": "2.0", "method": "Device.Panel", "Lock": "LOCK ALL", "Pass": "1234"}
```

PANEL GET LOCK SETTINGS

```
{"jsonrpc": "2.0", "method": "Device.GetPanel"}  
{"Lock": "UNLOCK ALL", "Pass": ""}
```

FINDER COMMANDS

Start/stop finder operation

```
{"jsonrpc": "2.0", "method": "Device.Finder", "Finder": true}
```

LCD FUNCTIONS

Print text on device frontal display. Two lines are available and aligned centre if Centre variable is true. Is possible to specify the display timeout in seconds.

```
{"jsonrpc": "2.0", "method": "Device.Print", "Line1": "Hi", "Line2": "Bye", "Center": true, "Timeout": 3}
```




All product characteristics are subject to variation due to production tolerances. **NEEC AUDIO BARCELONA S.L.** reserves the right to make changes or improvements in the design or manufacturing that may affect these product specifications.

For technical requests address to your supplier, distributor or fill the contact form in our website, at Support / Technical Request.

Motors, 166-168 08038 Barcelona - Spain - (+34) 932238403 | information@ecler.com | www.ecler.com