grass valley
A BELDEN BRAND

DENSITÉ/GECKOFLEX REST API

Reference Guide

M3059-0204-101

2018-02-16

www.grassvalley.com

Copyright and Trademark Notice

Terms and Conditions

Please read the following terms and conditions carefully. By using iControl Application Server documentation, you agree to the following terms and conditions.

Grass Valley hereby grants permission and license to owners of iControl Application Servers to use their product manuals for their own internal business use. Manuals for Grass Valley products may not be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose unless specifically authorized in writing by Grass Valley.

A Grass Valley manual may have been revised to reflect changes made to the product during its manufacturing life. Thus, different versions of a manual may exist for any given product. Care should be taken to ensure that one obtains the proper manual version for a specific product serial number.

Information in this document is subject to change without notice and does not represent a commitment on the part of Grass Valley.

Warranty information is available from the Legal Terms and Conditions section of Grass Valley's website (www.grassvalley.com).

Title	Densité/GeckoFlex REST API Reference Guide
Part Number	M3059-0204-101
Revision	2018-02-16, 16:31

toc

Table of Contents

1 Introduction

In iControl 7.20 and greater, Densité and GeckoFlex managers provide a REST service to monitor and control Densité and GeckoFlex cards.

By default, the REST service is activated with the following configuration:

- Available via HTTP on the AppServer
- Registered on port:
  - 5955 for Densité manager 1
  - 5953 for Densité manager 2
  - 5951 for Densité manager 3
  - 5949 for GeckoFlex
- With base path:
  - `http://10.0.3.6:5955/densite` for Densité managers
  - `http://10.0.3.6:5949/geckoflex` for GeckoFlex managers

For the purpose of this document:

- The IP `10.0.3.6` is used to represent the AppServer
- Densite manager 1 is used on URLs. For other Densité managers or for the GeckoFlex manager, use the proper base path and port number.

2 Service Description

This chapter provides the service description for the Densité or GeckoFlex REST Service.

Summary

Individual requests

Summary

Ping

Use this method to ensure that the REST service is running and accessible in the appropriate path.

Path	`http://10.0.3.6:5955/densite/ping`
Action	`GET`
Input	None
	Input format: N/A
Output	`pong`
	Output Media Type: text/plain

HTTP request example

```
GET /densite/ping HTTP/1.1
```

Host: `10.0.3.6:5955`

Cache-Control: no-cache

HTTP response example

pong

Get cards

Use this method to obtain the list of all accessible cards.

Path	`http://10.0.3.6:5955/densite/cards`
Action	GET
Headers	**Authorization**: Basic authorization
Input	None
	Input format: N/A
Output	CARD_LIST_JSON
	Output Media Type: application/json

HTTP request example

```
GET /densite/cards HTTP/1.1
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

Cache-Control: no-cache

HTTP response example

```
{"cards":[{"id":"AppServer_GV-Node_Densite_SLOT_17_177","name":"IFM-
2T","frameId":"AppServer_GV-
Node","slot":17,"devId":177,"version":"1.0.0"}]}
```

Get all parameter names

Use this method to obtain all the parameter names on a specific card.

Path	`http://10.37.94.20:5955/densite/cards/VM_DEV_MC_20_Densite_SLOT_2_64/parameters`
Action	GET
Headers	**Authorization**: Basic authorization
Input	**cardId**: The ID of the card to query
	Input format: In the path
Output	PARAMETER_JSON
	Output Media Type: application/json

HTTP request example

```
GET http://10.37.94.20:5955/densite/cards/VM_DEV_MC_20_Densite_SLOT_2_64/parameters
```

Authorization: Basic authorization

Cache-Control: no-cache

HTTP response example


```
{"cardId":"VM_DEV_MC_20_Densite_SLOT_2_64","parameters":{"ids":["aMuteAES6","aMuteAES5","aMuteAES8","aMuteAES7","aMuteAES3"]}}
```

Get one parameter

Use this method to obtain the current value of a specific parameter on a specific card.

Path	`http://10.0.3.6:5955/densite/cards/{cardId}/parameters/{parameterId}`
Action	GET
Headers	**Authorization**: Basic authorization
Input	**cardId**: The ID of the card to query
	Input format: In the path
Input	**parameterId**: The name of the parameter to get
	Input format: In the path
Output	PARAMETER_JSON
	Output Media Type: application/json

HTTP request example

```
GET /densite/cards/AppServer_GV-
Node_Densite_SLOT_17_177/parameters/PARAM::IFM.EthRtpIn[132].srcIpAddr HT
TP/1.1
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

Cache-Control: no-cache

HTTP response example

```
{"parameterId":"PARAM::IFM.EthRtpIn[132].srcIpAddr","value":[192,168,0,7]
}
```

Get parameter details

Use this method to obtain the parameter info for a specific parameter on a specific card.

Path	`http://10.0.3.6:5955/densite/cards/{cardId}/parameters/{parameterId}/info`
Action	GET
Headers	**Authorization**: Basic authorization
Input	**cardId**: The ID of the card to query
	Input format: In the path
Input	**parameterId:** The name of the parameter to get
	Input format: In the path
Output	parameterInfoDto
	Output Media Type: application/json

HTTP request example

```
GET
http://10.37.94.33:5955/densite/cards/MC_VM_33_MyFrame_Densite_SLOT_17_114/parameters/dLipsyncAud6StatusGRP/info
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

Cache-Control: no-cache

HTTP response example


```
{"choices":[{"index":"OFF","message":"OFF","active":true},{"index":"A","message":"A","active":true},
{"index":"B","message":"B","active":true}],"valueType":"String","parameterId":"dLipsyncAud6StatusGRP","name":"Group","active":true,"type":"choice","accessType":"Write only"}
```

Get many parameters

Use this method to obtain the current value of many parameters on a specific card.

Path	`http://10.0.3.6:5955/densite/cards/{cardId}/parameters/read`
Action	POST
Headers	**Authorization**: Basic authorization
	Content-Type: application/json
Input	**cardId**: The ID of the card to query **Input format**: In the path
Input	**Content**: LIST_OF_ID_JSON **Input format**: JSON payload in the body of the request
Output	LIST_OF_PARAMETER_JSON
	Output Media Type: application/json

HTTP request example

```
POST /densite/cards/AppServer_GV-Node_Densite_SLOT_17_177/parameters/read
HTTP/1.1
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

Content-Type: application/json

Cache-Control: no-cache

```
{"ids":["PARAM::IFM.EthRtpOut[25].destIpAddr",
"PARAM::IFM.EthRtpOut[25].destIpPort"]}
```

HTTP response example

```
{"parameters":[{"parameterId":"PARAM::IFM.EthRtpOut[25].destIpPort","valu
e":0},{"parameterId":"PARAM::IFM.EthRtpOut[25].destIpAddr","value":[0,0,0
,0]}]}
```

Set many parameters

Use this method to set the value of many parameters on a specific card.

Path	`http://10.0.3.6:5955/densite/cards/{cardId}/parameters/write`
Action	POST
Headers	**Authorization**: Basic authorization
	Content-Type: application/json
Input	**cardId**: The ID of the card to query
	Input format: In the path
Input	**Content**: LIST_OF_PARAMETER_JSON
	Input format: JSON payload in the body of the request
Output	HTTP Status 204 (No content)
	Output Media Type: N/A

HTTP request example

```
POST /densite/cards/AppServer_GV-
Node_Densite_SLOT_17_177/parameters/write HTTP/1.1
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

Content-Type: application/json

Cache-Control: no-cache


```
{"parameters":[{"parameterId":"PARAM::IFM.EthRtpOut[25].destIpPort","valu
e":0},{"parameterId":"PARAM::IFM.EthRtpOut[25].destIpAddr","value":[0,0,0
,0]}]}
```


HTTP response example

204: No Content

Session based requests

Sessions allow to receive notifications from changing parameters on specific cards. The content of the session output buffer is retrieve by a long-polling method. If the buffer contains data, this method returns immediately with the data. If not, then the call is held until data are acquired or a delay of 30 seconds has elapsed.

Summary

Connect

Use this method to create a new session and retrieve the session ID. Once it is created, the session output buffer contains the hardware information (available frames and cards).

> **Note:** A session timeout appears if the output buffer is not accessed for more than 40 seconds.

Path	`http://10.0.3.6:5955/densite/sessions/connect`
Action	POST
Headers	**Authorization**: Basic authorization
Input	Basic authorization
	Input format: N/A
Output	SESSION_ID_JSON
	Output Media Type: application/json

HTTP request example

```
POST /densite/sessions/connect HTTP/1.1
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

HTTP response example

```
{"id":"9b4ae4f9-9fe7-4a07-a0b8-5afc93b4c171"}
```

Disconnect

Use this method to terminate a session.

> **Note:** A session appears timeout appears if the output buffer is not accessed for more than 40 seconds.

Path	`http://10.0.3.6:5955/densite/sessions/{sessionId}`
Action	DELETE
Headers	**Authorization**: Basic authorization
Input	**sessionId**: The ID of the session
	Input format: In the path
Output	HTTP status 200 (Ok)
	Output Media Type: N/A

HTTP request example

```
DELETE /densite/sessions/9b4ae4f9-9fe7-4a07-a0b8-5afc93b4c171 HTTP/1.1
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

HTTP response example

Success 200: OK.

Session closed

Get session data

Use this method to retrieve the data available in a session. This method is held until data are available or until the maximum waiting time of 30 seconds is reached.

Path	`http://10.0.3.6:5955/densite/sessions/{sessionId}`
Action	GET
Headers	**Authorization**: Basic authorization
Input	**sessionId**: The ID of the session
	Input format: In the path
Output	HTTP status 304 (Not modified)
	Output Media Type: N/A
Output	SESSION_DATA_JSON
	Output Media Type:: application/json

HTTP request example

```
GET /densite/sessions/9b4ae4f9-9fe7-4a07-a0b8-5afc93b4c171 HTTP/1.
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

HTTP response example


```
{"hardware": [
   {"cardContainer": {"id": "Appserver_Gv-Node_Densite", "name": "Gv-
Node", "ipAddress": "10.0.3.7" }, "hardwareType": "card
container", "action": "added"},
   {"card": {"id": "Appserver_Gv-Node_Densite_SLOT_18_65530", "name":
"ETH3-REF", "frameId": "Appserver_Gv-Node_Densite", "slot": 18, "devId":
65530,"version": "1.0.2"}, "hardwareType": "card", "action": "added"},
   {"card": {"id": "Appserver_Gv-Node_Densite_SLOT_17_177", "name": "IFM-
2T", "frameId": "Appserver_Gv-Node_Densite", "slot": 17, "devId":
177, "version": "1.0.0"}, "hardwareType": "card", "action": "added"}
]}
```

Register card for notification

Use this method to register a card and receive notification when one of its parameters changes. Once the card is registered, the session output buffer contains all the current information of the card. This includes the list of parameters, the parameter information, and the parameter values).

Path	`http://10.0.3.6:5955/densite/sessions/{sessionId}/cards/{cardId}`
Action	PUT
Headers	**Authorization**: Basic authorization
Input	**sessionId**: The ID of the session
	Input format: In the path
Input	**cardId**: The ID of the card to register
	Input format: In the path
Output	`HTTP status 204` (No content)
	Output Media Type: N/A

HTTP request example

```
PUT /densite/sessions/9b4ae4f9-9fe7-4a07-a0b8-
5afc93b4c171/cards/Appserver_Gv-Node_Densite_SLOT_17_177 HTTP/1.1
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

HTTP response example

204: No Content

Unregister card from notification

Use this method to unregister a card in order to stop receiving notifications about it.

Path	`http://10.0.3.6:5955/densite/sessions/{sessionId}/cards/{cardId}`
Action	DELETE
Headers	**Authorization**: Basic authorization
Input	**sessionId**: The ID of the session
	Input format: In the path
Input	**cardId**: The ID of the card to register
	Input format: In the path
Output	`HTTP status 204` (No content)
	Output Media Type: N/A

HTTP request example

```
DELETE /densite/sessions/9b4ae4f9-9fe7-4a07-a0b8-
5afc93b4c171/cards/Appserver_Gv-Node_Densite_SLOT_17_177 HTTP/1.1
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

HTTP response example

204: No Content

Get one parameter

Use this method to request that the value of a parameter on a specific card be added to the session output buffer. This method differs from the individual request method. The result is returned asynchronously via the session output buffer rather than synchronously.

Path	`http://10.0.3.6:5955/densite/sessions/{sessionId}/cards/{cardId}/parameters/{parameterId}`
Action	GET
Headers	Authorization: Basic authorization
Input	**sessionId**: The ID of the session
	Input format: In the path
Input	**cardId**: The ID of the card to query
	Input format: In the path
Input	**parameterId**: The name of the parameter to get
	Input format: In the path
Output	`HTTP status 204` (No content)
	Output Media Type: N/A

HTTP request example

```
GET /densite/sessions/9b4ae4f9-9fe7-4a07-a0b8-
5afc93b4c171/cards/AppServer_GV-
Node_Densite_SLOT_17_177/parameters/PARAM::IFM.EthRtpIn[132].srcIpAddr
HTTP/1.1
```

Host: 10.0.3.6:5955

Authorization: Basic authorization

HTTP response example

204: No Content

3 JSON object description

Summary

CARD_LIST_JSON

```
{"cards":[CARD_JSON, ..., CARD_JSON]}
```

cards	An array of CARD_JSON

CARD_JSON

{"id": "AppServer_GV-Node_Densite_SLOT_2_168","name": "IPG-3901","frameId": "AppServer_GV-Node","slot": 12,"devId": 168,"version": "1.0.0"}

id	The ID of the card used to reference this card in the REST service
name	The name of the card. (i.e.: IPG-3901, MAP-3901, IFM-2T, etc...)
frameId	The ID of the frame that contains the card
slot	The number of the slot where the card is inserted
devId	The device ID that uniquely identifies the card type
version	The version of the firmware present on the card

PARAMETER_JSON

{"parameterId": "PARAM::PRESET.card.user[0].name","value": "CUSTOM"}

parameterId	The name of the parameter requested
value	The value of the parameter. This value can be a string, a boolean, a number, or an array.

LIST_OF_ID_JSON

```
{"ids":["PARAM::PRESET.card.user[0].name", ...,
"PARAM::PRESET.card.user[1].name"]}
```

ids	An array of the names of parameters to retrieve

LIST_OF_PARAMETER_JSON

```
{"parameters":[PARAMETER_JSON, ..., PARAMETER_JSON]}
```

SESSION_ID_JSON

```
{"id":"9b4ae4f9-9fe7-4a07-a0b8-5afc93b4c171"}
```

SESSION_DATA_JSON

```
{"hardware":[SESSION_DATA_HARDWARE_JSON, ...,
SESSION_DATA_HARDWARE_JSON],
 "values":[SESSION_DATA_VALUES_JSON, ..., SESSION_DATA_VALUES_JSON],
 "infos":[SESSION_DATA_INFOS_JSON, ..., SESSION_DATA_INFOS_JSON],
 "parameters":[SESSION_DATA_PARAMETER_LIST_JSON, ...,
SESSION_DATA_PARAMETER_LIST_JSON]}
```

SESSION_DATA_HARDWARE_JSON

SESSION_DATA_HARDWARE_CARD_CONTAINER_JSON or
SESSION_DATA_HARDWARE_CARD_JSON

SESSION_DATA_HARDWARE_CARD_CONTAINER_JSON

```
{"cardContainer": {"id": "Appserver_Gv-Node_Densite", "name":
"GvNode", "ipAddress": "10.0.3.7"}, "hardwareType": "card
container","action": "added"}
```

id	The ID that represents the frame. This is used to reference the frame in the REST service.
name	The name of the frame
ipAddress	The IP address of the frame
hardwareType	"card container" if the data represents a frame
action	"added" for a frame that is currently available
	"removed" for a frame that is no longer available

SESSION_DATA_HARDWARE_CARD_JSON

{"card": {"id": "Appserver_Gv-Node_Densite_SLOT_15_161", "name": "MAP-3901", "frameId": "Appserver_Gv-Node_Densite", "slot": 15, "devId": 161, "version": "1.0.0"}, "hardwareType": "card", "action": "added"}

id	The ID that represents the card
name	The name of the card. (i.e.: IPG-3901, MAP-3901, IFM-2T, etc...)
frameId	The ID of the frame that contains the card
slot	The number of the slot in which the card is inserted
devId	The device ID that uniquely identifies the card type
version	The version of the firmware present on the card
hardwareType	"card" if the data represents a card
action	"added" for a card that is currently available
	"removed" for a frame that is no longer available

SESSION_DATA_VALUES_JSON

```
{"cardId":"Appserver_Gv-Node_Densite_SLOT_15_161",
"parameter":PARAMETER_JSON}
```

cardId	The ID of the card
parameter	The json object that contains the parameterId and value

SESSION_DATA_INFOS_JSON

```
{"cardId":"Appserver_Gv-Node_Densite_SLOT_15_161",
"info":PARAMETER_INFO_JSON}
```

cardId	The ID of the card
info	The json object that contains the description of the parameter

SESSION_DATA_PARAMETER_LIST_JSON

```
{"cardId":"Appserver_Gv-Node_Densite_SLOT_15_161",
"parameters":{"ids":["PARAM::IFM.EthRtpIn[0].srcIpPort", ...,
"PARAM::IFM.EthRtpIn[143].srcIpPort"]}}
```

cardId	The ID of the card
ids:	An array that contains the name of all the parameters exposed by the card

4 List of parameters of supported cards

Title	Creator	Modified
IFM-2T	THOUIN Jean-Charles	Aug 26, 2016
IPG3901	THOUIN Jean-Charles	Aug 17, 2016

Grass Valley Technical Support

For technical assistance, contact our international support center, at 1-800-547-8949 (US and Canada) or +1-530-478-4148.

To obtain a local phone number for the support center nearest you, consult the Contact Us section of Grass Valley's website (www.grassvalley.com).

An online form for e-mail contact is also available from the website.

Corporate Head Office

Grass Valley
3499 Douglas-B.-Floreani
St-Laurent, Quebec H4S 2C6
Canada
Telephone: +1 514 333 1772
Fax: +1 514 333 9828
www.grassvalley.com