



# **NETAVIS Observer 4.4**

## **URL Command API**



# NETAVIS URL Command API for Observer 4.4

Document version V3 (Observer 4.4)

Published in June 2012

The software described in this manual is licensed under the terms of the NETAVIS end user license agreement and may only be used in accordance with these terms.

## Copyright

Copyright © 2003-2012 NETAVIS Software GmbH. All rights reserved.

NETAVIS is a trademark of NETAVIS Software GmbH.

NETAVIS Software GmbH

Blindengasse 3

A-1080 Vienna

Austria

Tel. +43 1 503 1722

Fax. +43 1 503 1722 360

[info@netavis.net](mailto:info@netavis.net)

[www.netavis.net](http://www.netavis.net)

# Contents

<b>1.</b>	<b>Introduction .....</b>	<b>4</b>
<b>2.</b>	<b>Live streaming .....</b>	<b>5</b>
2.1	Getting a live stream from a camera .....	5
<b>3.</b>	<b>Archive access .....</b>	<b>7</b>
3.1	Getting an archive stream from a recorded video .....	7
<b>4.</b>	<b>Receiving events .....</b>	<b>9</b>
4.1	Getting a live or an archive event stream .....	9
<b>5.</b>	<b>Control commands .....</b>	<b>12</b>
5.1	GetServerTime .....	12
5.2	OpenSession .....	12
5.3	CloseSession .....	13
5.4	OpenChannel .....	13
5.5	ReadChannel .....	14
5.6	CloseChannel .....	14
5.7	GetEntityTree .....	15
5.8	GetVideoArchiveMap .....	15
5.9	GetUserList .....	16
5.10	PropagateEvent .....	17
5.11	PerformAction .....	18
5.12	LiveSignal .....	18
5.13	VideoStreamAnnotation .....	19
5.14	ReadNextFrame .....	20
5.15	ReadNextEvent .....	20
5.16	RegisterCustomEvent .....	22
5.17	UnregisterCustomEvent .....	22
5.18	PropagateCustomEvent .....	23
5.19	SetArchiveProtection .....	23
5.20	GetUserLayouts .....	24
5.21	GetSmartGuardsOfWindow .....	25
5.22	GetPTZPositionList .....	25
5.23	GetPTZRoutesList .....	26
5.24	GetPTZRouteDefinition .....	27
5.25	LockPTZResource .....	28
5.26	RefreshPTZResource .....	28
5.27	ReleasePTZResource .....	28
5.28	StartPTZRoute .....	29
5.29	StopPTZRoute .....	29
5.30	SetPTZPosition .....	30
5.31	PTZCenterClick .....	30
5.32	ContinuousPTZ .....	31
5.33	ShowViewOfWindow .....	31
5.34	ShowCameraInViewport .....	32
5.35	StartSmartGuard .....	33
5.36	StopSmardGuard .....	33

---

# 1. Introduction

The NETAVIS URL Command API is a simple and effective way to interface a 3<sup>rd</sup> party application (e.g. a company's intranet/internet site) to a NETAVIS Observer video management system.

NETAVIS URL commands are simple HTTP URL requests and responses.

The URL Command API is also very suitable for the limited capabilities of mobile devices like smart phones, PDAs, and tablets. For example, live and archive video streams are delivered as multipart HTTP streams that are well supported by such devices.

## Format of an URL command

A NETAVIS URL command looks like a standard HTTP request:

**http://192.168.7.221/urlapi/live?SessionID=1** where

- http://192.168.7.221 is the server root context
- /urlapi/live is the resource base
- ? is the parameter separator (or & for the following parameters)
- SessionID=1 is the parameter part

The server root context may vary, depending on routers, firewalls or other user specific or environment setup conditions. The server ip/name may be followed by a colon and a port number.

The Netavis URL commands are grouped by resource bases:

- Live streaming
- Archive streaming
- Event streaming
- Control query commands

Resources and parameters are case sensitive.

## 2. Live streaming

### 2.1 Getting a live stream from a camera

Resource base: /urlapi/live  
 Response Mime-Types: **image/jpeg** for snapshots and  
**multipart/x-mixed-replace** for streams  
**application/json** for errors

Parameter	Group	Description	Default Values
User	Access	The name of a valid Netavis User	default: 'guest'
Passwd		The MD5 hash coded value of the password	default: MD5('guest')
SessionID	Access	Optionally, it is also available to use an open session for live streaming via its SessionID. For security reason the open session identified by its SessionID available only from the same IP where it had been E.g. the OpenSession command of JSON request ??? is it session OR user/pwd or can you use both together?	
EntityID		Unique ID of the camera	mandatory
Size	Format	One of: Small, Medium, Large or WxH. WxH is width and height in pixels. The server tries to find and use the closest resolution to the given WxH.	default: Large
Quality		One of: Low, Medium, High	default: High
Fps		Frame per second	default: 5
Type		One of: JPEG, MPEG4, MxPEG, H264	default: JPEG
ProfileName	Profile	The name of the profile belongs to the given CameraID. To use profile-based stream identification, the available profiles of the camera must be known. For getting the available stream profiles have a look at the GetStreamProfileList command.	
ProfileID		The unique identifier of the stream belongs to the give CameraID	
Snapshot		Use when a snapshot is need from the	default: false

Parameter	Group	Description	Default Values
		camera only	
DoNotCloseSession		True if the session is also used somewhere else (E.g. further usage)	default: false

## Examples

1. Requesting for a live 'Snapshot' of camera '5' as 'guest' user  
<http://192.168.7.221/urLapi/Live?EntityID=5&Snapshot=true>
2. Requesting for a live stream with the user named 'admin' in a multipart motion JPEG stream from the camera '38' with the encoded password of 'admin' ('21232f297a57a5a743894a0e4a801fc3') in 'High' quality 'Medium' size and in '5' Fps:  
<http://192.168.7.221/urLapi/Live?User=admin&Passwd=21232f297a57a5a743894a0e4a801fc3&EntityID=38&Size=Medium&Quality=High&Fps=5&Type=JPEG>
3. Request via encrypted credentials:  
<http://192.168.7.221/urLapi/Live?Login=Xx1D67xZKVYtgA2mWh6TJxBLkD5Jek%2FVrouDkScNiN4%3D&EntityID=38&Size=Medium>
4. Request for the same live stream using the earlier given SessionID '1':  
<http://192.168.7.221/urLapi/Live?SessionID=1&EntityID=38&Size=Medium>
5. Request for the live stream identified by its profile name:  
[http://192.168.7.221/urLapi/Live?SessionID=1&EntityID=38&ProfileName=VGA\\_JPG](http://192.168.7.221/urLapi/Live?SessionID=1&EntityID=38&ProfileName=VGA_JPG)
6. A positive multipart HTTP sample response:
 

```
HTTP/1.0 200 OK\r\n
Content-Type: multipart/x-mixed-replace; boundary=--boundary\r\n
\r\n
--boundary\r\n
Content-Type: image/jpeg\r\n
Content-Length: <image size>\r\n
\r\n
<JPEG image data>\r\n
--boundary\r\n
...
```
7. Negative "application/json" mime-type sample response :
 

```
{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Illegal user/passwd. User=admin1",
  "ReturnCode": 103
}}}
```

## 3. Archive access

### 3.1 Getting an archive stream from a recorded video

Resource base: /urlapi/archive

Response Mime-Types: **image/jpeg** for snapshots and **multipart/x-mixed-replace** for streams  
**application/json** for errors

Parameter		Description	Default Values
User	Access	The name of a valid Netavis User	default: 'guest'
Passwd		The MD5 hash coded value of the password	default: MD5('guest')
SessionID	Access	Optionally, it is also available to use an open session for live streaming via its SessionID. For security reason the open session identified by its SessionID available only from the same IP where it had been E.g. the OpenSession command of JSON request ??? is it session OR user/pwd or can you use both together?	
EntityID		Unique ID of the camera	mandatory
FrameStep		Streams only every FrameStep images	default: 0
MaxFrames		Max. number of frames to stream	default: 0
Backward		Backward direction of archive streaming	default: false
Type		Image type to retrieve. One of: JPEG, MPEG4, MxPEG, H264	default: JPEG
Snapshot		Use when a snapshot is need from the archive only	default: false
ControlledBy		One of: USER, SERVER. ControlledBy is an optional parameter. For the default USER controlled stream there must be another URL Command requester to ask the server to put the next archive image into the open archive stream. For a SERVER controlled archive stream the server manages the frames putting into the stream via the frames' timestamp. For further information see the Control commands, as well	default: SERVER

Parameter			Description	Default Values
DoNotCloseSession			True if the session is also used somewhere else (E.g. further usage)	default: false
Filter	DateTimeFilter	Start	Timestamp in human readable format to start the streaming from	default: none
		End	Timestamp in human readable format to finish streaming at	
		Duration	Duration to streaming until	
	MillisFilter	Start	Timestamp in epoch format to start the streaming from	
		End	Timestamp in epoch format to finish streaming at	
		Duration	Duration to streaming until	

Explanation of parameters:

DateTimeFilter or MillisFilter are only parameter values of TimeFilter. Without it the archive stream starts from the last recent available and plays until the first recent available images in forward direction. In background direction it plays from the first to the last recent available images. DateTimeFilter is different from MillisFilter only the format; while DateTimeFilter uses human readable form, the MillisFilter uses epoch format.

## Examples

1. Requesting for the most recent archive 'Snapshot' image of camera '5' as 'guest' user.  
<http://192.168.7.221/urllapi/archive?EntityID=5&Snapshot=true>  
 Note: Without user and passwd parameters the default (guest/guest) will be used.
2. Requesting for a server controlled playback archive stream as user 'admin'/'admin' start from 2011-10-18 10:00:00.000 from the camera '38':  
<http://192.168.7.221/urllapi/archive?User=admin&Passwd=21232f297a57a5a743894a0e4a801fc3&EntityID=38&Filter=DateTimeFilter&Start=20111018T10:00:00.000>
3. Requesting for the same archive stream as before using the earlier given SessionID '1':  
<http://192.168.7.221/urllapi/archive?SessionID=1&EntityID=38&Filter=DateTimeFilter&Start=20111018T10:00:00.000>

4. A positive snapshot sample response:

```
HTTP/1.0 200 OK\r\n
Content-Type: image/jpeg\r\n
Content-Length: <image size>\r\n\r\n
<JPEG image data>\r\n
```

5. Negative response sample as "application/json" mime-type:

```
{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Illegal user/passwd. User=admin1",
  "ReturnCode": 103}}}
```



## 4. Receiving events

### 4.1 Getting a live or an archive event stream

Resource base: /urlapi/event

Response Mime-Types: **multipart/x-mixed-replace** with **application/json** contents or simple **application/json** for errors

Parameter			Description	Default Values
User		Access	The name of a valid Netavis User	default: 'guest'
Passwd			The MD5 hash coded value of the password	default: MD5('guest' )
SessionID		Access	Optionally, it is also available to use an open session for live streaming via its SessionID. For security reason the open session identified by its SessionID available only from the same IP where it had been E.g. the OpenSession command of JSON request ??? is it session OR user/pwd or can you use both together?	
Mode			One of { Live, Archive, ArchiveAndLive }	default: Live
EventType			Event type to filter, or none for all	default: none
MaxEvents			The number of max events to retrieve 0 means no max filter	default for none Live is 50 for rest is none
PropertyFilterNameX		Property Filter	The name of the event property to filter on	default: none
PropertyFilterValueX			The value of the event property to filter on	
Filter	DateTimeFilter	Start	Timestamp in human readable format to start the streaming from	default: none
		End	Timestamp in human readable format to finish streaming at	
		Duration	Duration to streaming until	
	MillisTimeFilter	Start	Timestamp in epoch format to start the streaming from	

Parameter			Description	Default Values
		End	Timestamp in epoch format to finish streaming at	
		Duration	Duration to streaming until	
Backward			Backward direction on an archive transmission	default: false
Snapshot			Use when the very last 50 available events are interested	default: false
DoNotCloseSession			True if the session is also used somewhere else (E.g. further usage)	default: false
SendEventsSeparately			True if the stream is manually controlled	default: false

Explanation of parameters:

PropertyFilterName should be followed by an index from 0 to define a PropertyFilter together with the same indexed PropertyFilterValue

PropertyFilter is not a parameter it symbolizes only the usage of PropertyFilterName & PropertyFilterValue together

Filter is an optional parameter and not a symbol. DateTimeFilter or MillisTimeFilter are only parameter values of TimeFilter. Without it the archive stream starts from the last recent available and plays until the first recent available images in forward direction. In background direction it plays from the first to the last recent available images. DateTimeFilter is different from MillisTimeFilter only the format; while DateTimeFilter uses human readable form, the MillisTimeFilter uses epoch format.

Backward is available only in Archive event stream

Only the Access, MaxEvents & DoNotCloseSession parameters are used if Snapshot has been defined as true

SendEventsSeparately is a secure way of receiving all events of a stream with server side care of those events what arrives in time when the client is not listening on the channel. The server replies to SendEventsSeparately a StreamID and a ChannelID belongs to the newly created event stream what needs to be used for ReadNextEvent call via a new control query. See the documentation of the ReadNextEvent command in the next paragraph, too.

## Examples

1. Request for the last 50 live events:  
<http://192.168.7.221/urLapi/event?SessionID=1&Snapshot=true>
2. A positive multipart HTTP sample response:

```
HTTP/1.0 200 OK\r\n
Content-Type: multipart/x-mixed-replace;boundary=--boundary\r\n
\r\n
--boundary\r\n
Content-Type: application/json\r\n
Content-Length: <json event size>\r\n\r\n
<json event data>\r\n
--boundary\r\n
...
```

3. Negative "application/json" mime-type sample response :

```
{"SNAP": {"ExecutionStatus": {  
  "ErrorText": "Illegal user/passwd. User=admin1",  
  "ReturnCode": 103  
}}}
```

4. A positive response to SendEventsSeparately=true kind event setup:

```
{"SNAP":{"EventStream":{"SessionID":4823,"LiveStream":{"StreamID":1,"MaxEvents":0},"ChannelID":1}}}
```

## 5. Control commands

Resource base: `/urlapi/control`

Response Mime-Types: ***application/json***

Every Control request has a mandatory parameter named **Command** in the first position of the URL parameters. The different commands may have further mandatory or optional parameters what are described in the following. In the description of the next URL commands we hide the mandatory parameter **COMMAND** and list only the other parameters. Other words, every Control query URL Commands must start with the URL Base right after followed by the parameter **Command=<COMMAND>**. Please have a look at the examples.

### 5.1 GetServerTime

Used for getting the current time of the server in milliseconds, UNIX epoch time format. This command has no other parameters.

#### Example

- Sample GetServerTime command request:  
`http://192.168.7.221/urlapi/control?Command=GetServerTime`
- Positive response to an OpenSession command:  
`{"SNAP": {"ServerTime": 1324481877510}}`
- Negative response to an OpenSession command:  
`{"SNAP": {"ExecutionStatus": {"ErrorText": "No Command: 'GetServerTimestmap' matched"}}`

### 5.2 OpenSession

Used for opening a communication tunnel, identified by the user credentials to send further requests to or retrieve data from the server. Every session has its own and unique SessionID produced by the server in the reply to the OpenSession request command. A session belongs to its user with the IP address of the remote client machine where it was opened. Other clients from different IP address can not use the open session at all. In the other hand an open session can also use to handle requests of the live or archive resources, just sending its SessionID instead of the credential parameters to in the live or archive request. Because this session is really a SNAP session it is also important to handle an open session by the SNAP requirements E.g. periodically sending LiveSignal requests to keep it open if no channel was open on it. At the end of the communication, it is highly recommended to close the session, too.

Parameter		Description	Default Values
User	Access	The name of a valid Netavis user	default: 'guest'
Passwd		The MD5 hash coded value of the password	default: MD5('guest')

#### Example

- Sample OpenSession command request:

<http://192.168.7.221/urlapi/control?Command=OpenSession&User=admin&Passwd=21232f297a57a5a743894a0e4a801fc3>

- Positive response to an OpenSession command:  

```
{ "SNAP": { "NewSession": { "UID": 1, "SessionID": 4332 } } }
```
- Negative response to an OpenSession command:  

```
{ "SNAP": { "ExecutionStatus": {  
  "ErrorText": "Session not found. SessionID=4325",  
  "ReturnCode": 200  
} } }
```

## 5.3 CloseSession

Used for closing an earlier opened session. It also closes all related open channels, live or archive streams of the session. It also frees all resources on the server side.

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory

### Example

- Sample request for closing an open session:  
<http://192.168.7.221/urlapi/control?Command=CloseSession&SessionID=4325>
- Response of the CloseSession command:  

```
{ "SNAP": { "ExecutionStatus": { "ReturnCode": 0 } } }
```
- Negative response to CloseSession command:  

```
{ "SNAP": { "ExecutionStatus": {  
  "ErrorText": "Session not found. SessionID=4325",  
  "ReturnCode": 200  
} } }
```

## 5.4 OpenChannel

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
TimeLimit	The camera identifier	default: 0
DataLimit	One of: Minute, Day	default: 0
FrameBased	Streaming without GOP	default: false

### Example

- Sample request for getting a new channel:  
<http://192.168.7.221/urlapi/control?Command=OpenChannel&SessionID=4325>
- Positive response to the OpenChannel command:  

```
{ "SNAP": { "NewChannel": { "ChannelID": 1 } } }
```
- Negative response to OpenChannel command:

```
{"SNAP": {"ExecutionStatus": {  
  "ErrorText": "Session not found. SessionID=4325",  
  "ReturnCode": 200  
}}}
```

## 5.5 ReadChannel

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
ChannelID	Channel identifier given from OpenChannel	mandatory
ForceSingleFrame	True if only one frame is needed	default: true

Note: for getting any content via ReadChannel Command you have to setup the Channel before start to read. E.g. you have to subscribe to retrieve event notifications of an event type. Please read the related SNAP ReadChannel function how to use ReadChannel and for further information.

### Example

- Sample request for reading the next frame from an open channel:  
<http://192.168.7.221/urLapi/control?Command=ReadChannel&SessionID=4325&ChannelID=1&ForceSingleFrame=true>
- Positive response to the ReadChannel command:
- Negative response to ReadChannel command:

```
{"SNAP": {"ExecutionStatus": {  
  "ErrorText": "Session not found. SessionID=4325",  
  "ReturnCode": 200  
}}}
```

## 5.6 CloseChannel

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
ChannelID	Channel identifier given from OpenChannel	mandatory

### Example

- Sample request for closing an open channel:  
<http://192.168.7.221/urLapi/control?Command=CloseChannel&SessionID=4325&ChannelID=1>
- Positive response to the CloseChannel command:  
*{"SNAP": {"ExecutionStatus": {"ReturnCode": 0}}}*
- Negative response to CloseChannel command:

```
{"SNAP": {"ExecutionStatus": {
"ErrorText": "Session not found. SessionID=4325",
"ReturnCode": 200
}}}
```

## 5.7 GetEntityTree

Used for retrieving available entities and its hierarchy from the server. Optionally the root point of the tree can define via the RootEntityID, without it the complete entity tree are going to be send.

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
RootEntityID	The ID of the root entity to start the tree from	default: 0
RequestedDetails	Comma separated list of detail descriptors. One or list of: IP_ADDRESS, STATUS_CODES, PTZ_DETAILS	optional

### Example

- Sample request for a GetEntityTree command:  
[http://192.168.7.221/urllapi/control?Command=GetEntityTree&SessionID=4325&RequestedDetails=PTZ\\_DETAILS](http://192.168.7.221/urllapi/control?Command=GetEntityTree&SessionID=4325&RequestedDetails=PTZ_DETAILS)
- Positive Response to GetEntityTree command:  

```
{"SNAP": {"EntityTree": {"EntityTreeNode": [
{
"Host": {
"HostID": 13792383803876560,
"Address": "127.0.0.1",
"Connected": true,
"Name": "netavis"
},
"ParentEntityID": 0,
"Type": "Group",
"Name": "cust1 Cameras",
"EntityID": 1
}
]}}}
```
- Negative response to GetEntityTree command:  

```
{"SNAP": {"ExecutionStatus": {
"ErrorText": "Session not found. SessionID=4325",
"ReturnCode": 200 }}}
```

## 5.8 GetVideoArchiveMap

Used for retrieving the map of the available video records from the server. It can filter for a camera or a camera group, for the start timestamp and can change its resolution by the time length.

Parameter	Description	Default Values
-----------	-------------	----------------

Parameter		Description		Default Values
SessionID		The session identifier given from OpenSession		mandatory
CameraID		The camera identifier		mandatory
Unit		One of: Minute, Day		default: Minute
Filter	DateTimeFilter	Start	Timestamp in human readable format to start the streaming from	
		End	Timestamp in human readable format to finish streaming at	
		Duration	Duration to streaming until	
	MillisFilter	Start	Timestamp in epoch format to start the streaming from	
		End	Timestamp in epoch format to finish streaming at	
		Duration	Duration to streaming until	

Explanation of parameters:

DateTimeFilter or MillisFilter are only parameter values of TimeFilter. Without it the archive stream starts from the last recent available and plays until the first recent available images in forward direction. In background direction it plays from the first to the last recent available images. DateTimeFilter is different from MillisFilter only the format; DateTimeFilter uses human readable form, the MillisFilter uses epoch format.

### Example

- Sample request for getting an archive video map:  
<http://192.168.7.221/urlapi/control?Command=GetVideoArchiveMap&SessionID=4325&CameraID=38>
- Positive response to the GetVideoArchiveMap command:

```
{"SNAP": {"VideoArchiveMap": {  
    "Unit": "Minute",  
    "StartDateTime": "2011-10-07T00:00:00.000+02:00",  
    "EntityID": 38,  
    "MapItem": [ { "FrameCount": 1800, "content":  
        "0000000000000000000000000000000011111111111111111111111111111111" } ] }}
```
- Negative response to GetVideoArchiveMap command:

```
{"SNAP": {"ExecutionStatus": {  
    "ErrorText": "Session not found. SessionID=4325",  
    "ReturnCode": 200  
}}}
```

## 5.9 GetUserList

Parameter	Description	Default Values
SessionID	The session identifier given from	mandatory



Parameter	Description	Default Values
	OpenSession	
UserD	Unique user identifier	mandatory Non-zero value filters one single user, zero value returns all users

## Example

- Sample request for getting list of users  
<http://192.168.7.221/urLapi/control?Command=GetUserList&SessionID=4255&UserID=0>
- Positive response to the GetUserList command:
 

```
{
  "SNAP": {
    "UserList": [
      {
        "UserID": 1,
        "Name": "admin",
        "LoginName": "Admin User",
        "GroupID": 1,
        "CustomerID": 1,
        ...
      },
      {
        "UserID": 2,
        "Name": "Guest User",
        ...
      }
    ]
  }
}
```
- Negative response to GetUserList command:
 

```
{
  "SNAP": {
    "ExecutionStatus": {
      "ErrorText": "Session not found. SessionID=4325",
      "ReturnCode": 200
    }
  }
}
```

## 5.10 PropagateEvent

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
EventType	Type of the Event	mandatory
EventPropertyName[x]	Name of the [x] event property	
EventPropertyValue[x]	Value of the [x] event property	
EventPropertyType[x]	Type of the [x] event property	

## Example

- Sample request for propagating a new event:

<http://192.168.7.221/urLapi/control?Command=PropagateEvent&SessionID=4325&Event Type=MotionDetection&EventPropertyName0=CameraID&EventPropertyType0=Integer&EventPropertyValue0=4>

- Positive response to the PropagateEvent command:  
{"SNAP": {"ExecutionStatus": {"ReturnCode": 0}}}
- Negative response to PropagateEvent command:

```

{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Session not found. SessionID=4325",
  "ReturnCode": 200
}}}
```

## 5.11 PerformAction

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
ActionType	Type of the Event	mandatory
ActionPropertyName[x]	Name of the [x] action property	
ActionPropertyValue[x]	Value of the [x] action property	
ActionPropertyType[x]	Type of the [x] action property	

### Example

- Sample request for performing an action on the server:  
<http://192.168.7.221/urLapi/control?Command=PerformAction&ActionType=SetCameraPTZPosition&SessionID=4325&ActionPropertyName0=CameraID&ActionPropertyType0=Integer&ActionPropertyValue0=1&ActionPropertyName1=PositionName&ActionPropertyType1=String&ActionPropertyValue1=Door>
- Positive response to the PerformAction command:  
{"SNAP": {"ExecutionStatus": {"ReturnCode": 0}}}
- Negative response to PerformAction command:

```

{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Session not found. SessionID=4325",
  "ReturnCode": 200
}}}
```

## 5.12 LiveSignal

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	default: 0

## Example

- Sample request for sending a live signal:  
<http://192.168.7.221/urLapi/control?Command=LiveSignal&SessionID=4325>
- Positive response to the LiveSignal command:  

```
{"SNAP": {"ExecutionStatus": {"ReturnCode": 0}}}
```
- Negative response to LiveSignal command:  

```
{"SNAP": {"ExecutionStatus": {"ErrorText": "Session not found. SessionID=4325", "ReturnCode": 200}}}
```

## 5.13 VideoStreamAnnotation

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
CameraID	ID of a camera or camera group	mandatory
Text	Text to display on camera view	
Action	One of: Start, Stop	default: Start
Destination	One of: Live, Archive, LiveAndArchive	default: Live
StartTimestamp	Epoch or human readable timestamp format of annotation starting	default: <now>
StopTimestamp	Epoch or human readable timestamp format of annotation stopping	
StopAfterMillisec	Millisec value of stoppping after start	default: 10 sesc
EventStorage	One of: UnSave, SaveStart, SaveStop, SaveAll	default: SaveStart
EventVisibility	One of: Hide, ShowStart, ShowStop, ShowAll	default: Hide
ForegroundColor		
BackgroundColor		
FontName		
FontSize		
Alignment		
Margin		
Wrapping		
StopOnClick		
BlinkDuration		

Note: The parameters from ForegroundColor to BlinkDuration are not yet supported on server-side

## Example

- Sample request for sending a live VideoStreamAnnotation:  
<http://192.168.7.221/urLapi/control?Command=VideoStreamAnnotation&SessionID=4325&CameraID=1&Text=AnnotationMessage>  
 Note: Do not forget to encode the url before send it like this:  
 http%3A%2F%2F192.168.7.221%2Fmobile%2Fcontrol%3FCommand%3DVideoStreamAnnotation%26SessionID%3D42234%26CameraID%3D1%26Text%3DAnnotation+message
- Positive response to the VideoStreamAnnotation command:  

```
{ "SNAP": { "ExecutionStatus": { "ReturnCode": 0 } } }
```
- Negative response to VideoStreamAnnotation command:  

```
{ "SNAP": { "ExecutionStatus": { "ErrorText": "Session not found. SessionID=4325", "ReturnCode": 200 } } }
```

## 5.14 ReadNextFrame

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
ChannelID	Channel identifier given from stream query setup	mandatory
StreamID	Stream identifier given from stream query setup	mandatory

Note: To use ReadNextFrame it must be setup the stream first via a stream setup query with a ControlledBy=USER parameter. See the documentation of stream query command, too

## Example

- Sample request for reading the next event from an open channel:  
<http://192.168.7.221/urLapi/control?Command=ReadNextFrame&SessionID=4325&ChannelID=1&StreamID=1>
- Positive response ReadNextFrame command:
- Negative response to ReadNextFrame command:  

```
{ "SNAP": { "ExecutionStatus": { "ErrorText": "Session not found. SessionID=4325", "ReturnCode": 200 } } }
```

## 5.15 ReadNextEvent

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
ChannelID	Channel identifier given from event query setup	mandatory

Note: To use ReadNextEvent it must be setup the event stream first via an event query with a SendEventsSeparately=true parameter. See the documentation of event query command, too.

## Example

- Sample request for reading the next event from an open channel:  
<http://192.168.7.221/urLapi/control?Command=ReadNextEvent&SessionID=4325&ChannelID=1>

- Positive response ReadNextEvent command:

```
{
  "SNAP": {
    "Event": {
      "EventType": "MotionDetection",
      "EventID": 89009793,
      "DateTimeStamp": "2012-01-05T14:03:35.136+01:00",
      "MillisStamp": 1325768615136,
      "EventProperty": [
        {
          "Value": "",
          "Type": "String",
          "Name": "EventText"
        },
        {
          "Value": "Motion event forgalom on camera cust1:4.120 ().",
          "Type": "String",
          "Name": "EventText"
        },
        {
          "Value": 7,
          "Type": "Integer",
          "Name": "ClassID"
        },
        {
          "Value": 38,
          "Type": "Integer",
          "Name": "CameraID"
        },
        {
          "Value": "cust1:4.120",
          "Type": "String",
          "Name": "CameraName"
        },
        {
          "Value": "forgalom",
          "Type": "String",
          "Name": "EventName"
        },
        {
          "Value": "",
          "Type": "String",
          "Name": "EventComment"
        },
        {
          "Value": 5564921012190495,
          "Type": "Long",
          "Name": "MDParams"
        },
        {
          "Value": 100,
```

```

        "Type": "Integer",
        "Name": "Priority"
    },
    {
        "Value": 1325768615136,
        "Type": "Long",
        "Name": "EventStamp"
    },
    ...
]
}}}

```

- Negative response to ReadNextEvent command:

```

{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Session not found. SessionID=4325",
  "ReturnCode": 200
}}}

```

## 5.16 RegisterCustomEvent

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
EventPropertyName[x]	Name of the [x] event property	

### Example

- Sample request for registering a new custom event:  
<http://192.168.7.221/urLapi/control?Command=RegisterCustomEvent&SessionID=4325&EventPropertyName0=Room>

- Positive response to the RegisterCustomEvent command:  

```
{"SNAP": {"ExecutionStatus": {"ReturnCode": 0}}}
```
- Negative response to RegisterCustomEvent command:

```

{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Session not found. SessionID=4325",
  "ReturnCode": 200
}}}

```

## 5.17 UnregisterCustomEvent

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory

### Example

- Sample request for unregistering a custom event:

<http://192.168.7.221/urLapi/control?Command=UnregisterCustomEvent&SessionID=4325>

- Positive response to the UnregisterCustomEvent command:
- Negative response to UnregisterCustomEvent command:

```
{"SNAP": {"ExecutionStatus": {
"ErrorText": "Session not found. SessionID=4325",
"ReturnCode": 200
}}}
```

## 5.18 PropagateCustomEvent

Parameter	Description	Default Values
SessionID	The session identifier given from OpenSession	mandatory
EventPropertyName[x]	Name of the [x] event property	
EventPropertyValue[x]	Value of the [x] event property	
EventPropertyType[x]	Type of the [x] event property	

### Example

- Sample request for propagating a new custom event:  
<http://192.168.7.221/urLapi/control?Command=PropagateCutomEvent&SessionID=4325&EventPropertyName0=CameraID&EventPropertyType0=Integer&EventPropertyValue0=4>
  - Positive response to the PropagateCustomEvent command:
  - Negative response to PropagateCustomEvent command:
- ```
{"SNAP": {"ExecutionStatus": {
"ErrorText": "Session not found. SessionID=4325",
"ReturnCode": 200
}}}
```

## 5.19 SetArchiveProtection

| Parameter | Description                                       | Default Values                        |
|-----------|---------------------------------------------------|---------------------------------------|
| SessionID | The session identifier given from OpenSession     | mandatory                             |
| EntityID  | Unique camera identifier                          | mandatory                             |
| Set       | Protect or unprotect the selected interval        | Mandatory, one of:<br>{ true, false } |
| From      | Timestamp in format of<br>yyyy-MM-ddTHH:mm:ss.SSS | mandatory                             |

| Parameter | Description                                       | Default Values |
|-----------|---------------------------------------------------|----------------|
| To        | Timestamp in format of<br>yyyy-MM-ddTHH:mm:ss.SSS | mandatory      |

## Example

- Sample request for protecting archive records of camera #1 from noon to one o'clock:  
<http://192.168.7.221/urLapi/control?Command=SetArchiveProtection&SessionID=4255&EntityID=1&Set=true&From=2011-12-14'T'12:00:00.000&To=2011-12-14'T'13:00:00.000>
- Positive response to the SetArchiveProtection command:  

```
{"SNAP": {"ExecutionStatus": {"ReturnCode": 0}}}
```
- Negative response to SetArchiveProtection command:  

```
{"SNAP": {"ExecutionStatus": {"ErrorText": "Session not found. SessionID=4325", "ReturnCode": 200}}}
```

## 5.20 GetUserLayouts

| Parameter | Description                                   | Default Values |
|-----------|-----------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession | mandatory      |
| UserID    | User identifier                               | mandatory      |

## Example

- Sample request for getting all layouts of user #1:  
<http://192.168.7.221/urLapi/control?Command=GetUserLayout&SessionID=4255&UserID=1>
- Positive response to the command:  

```
{"SNAP": {"UserLayouts": {"Windows": {"Window": [{"ViewGroups": {"ViewGroup": {"ViewGroupElements": {"ViewGroupElement": [{"Duration": 10, "ViewGroupID": 3, "ElementID": 1, "PanelNumber": 1}, {"Duration": 10, "ViewGroupID": 3, "ElementID": 2, "PanelNumber": 2}], "count": 2}, {"ViewGroupName": "nana", "ViewGroupID": 3, "UserID": 1, "WindowID": 3}, {"count": 1}, {"WindowName": "Window 3", "WindowID": 3, "Views": {"View": {"UserID": 1, "Flags": 1, "PanelNr": 1, "WindowID": 3, "ViewPorts": {"ViewPort": {"PortEntities": {"PortEntity": [{"PortNr": 1, "Flags": 4101, "EntityNr": 0, "Cameras": {"Camera": {"Name": "cust1 Cam1", "EntityID": 3}, {"count": 1}, {"Info": "crp1=-1, crp0=-1, isize=18", "EntityID": 3}, {"PortNr": 1, "Flags": 4100, "EntityNr": 1, "Cameras": {"Camera": {"Name": "7.92", "EntityID": 37}, {"count": 1}, {"Info": "crp1=-1, crp0=-1, isize=21", "EntityID": 37}], "count": 2}, {"PortNr": 1, "Flags": 6, "PanelNr": 1, "Info": "zm-aids, prefType=2, crop=0, strech=1"}, {"count": 1}, {"PanelType": {"AspectRatio": "Fill", "ViewIndexName": "Matrix-1S", "ViewIndex": 13}, {"Info": "car=0.0", "PanelName": 1}, {"count": 1}}], "count": 1}}], "count": 1}}], "count": 1}}}
```
- Negative response to the command:



```
{"SNAP": {"ExecutionStatus": {
"ErrorText": "Session not found. SessionID=4325",
"ReturnCode": 200
}}}
```

## 5.21 GetSmartGuardsOfWindow

| Parameter | Description                                                 | Default Values |
|-----------|-------------------------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession               | mandatory      |
| UserID    | User identifier for which the guards are requested          | mandatory      |
| WindowID  | Identifier of the window for which the guards are requested | mandatory      |

### Example

- Sample request for getting the guard list for user 1, window 1  
<http://192.168.7.221/urLapi/control?Command=GetSmartGuardsOfWindow&SessionID=4255&UserID=1&WindowID=1>
- Positive response to the command:  

```
{"SNAP":{"ViewGroups":{"ViewGroup":{"ViewGroupElement":[{"Duration":10,"ViewGroupID":1,"ElementID":1,"PanelNumber":99996},{"Duration":10,"ViewGroupID":1,"ElementID":2,"PanelNumber":5},{"Duration":10,"ViewGroupID":1,"ElementID":3,"PanelNumber":4},{"Duration":10,"ViewGroupID":1,"ElementID":4,"PanelNumber":3},{"Duration":10,"ViewGroupID":1,"ElementID":5,"PanelNumber":2},{"Duration":10,"ViewGroupID":1,"ElementID":6,"PanelNumber":1}],"ViewGroupName":"gray","ViewGroupID":1,"UserID":1,"WindowID":1}}}}
```
- Negative response to the command:  

```
{"SNAP": {"ExecutionStatus": {
"ErrorText": "Session not found. SessionID=4325",
"ReturnCode": 200
}}}
```

## 5.22 GetPTZPositionList

| Parameter | Description                                   | Default Values |
|-----------|-----------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession | mandatory      |
| EntityID  | Unique camera identifier                      | mandatory      |

### Example

- Sample request for getting a list of ptz positions of camera #38.  
<http://192.168.7.221/urLapi/control?Command=GetPTZPositionList&SessionID=4255&EntityID=38>

- Positive response to the GetPTZPositionList command:

```
{
  "SNAP": {
    "PTZPositionList": {
      "PTZPosition": [
        {
          "PresetPosID": 1,
          "Zoom": 7201,
          "Name": "kapu",
          "Tilt": -1246,
          "Pan": -1979,
          "EntityID": 37,
          "Comment": ""
        },
        {
          "PresetPosID": 2,
          "Zoom": 7201,
          "Name": "ház",
          "Tilt": -1011,
          "Pan": -2243,
          "EntityID": 37,
          "Comment": ""
        },
        {
          "PresetPosID": 3,
          "Zoom": 7201,
          "Name": "épület",
          "Tilt": -1006,
          "Pan": -2537,
          "EntityID": 37,
          "Comment": ""
        }
      ]
    }
  }
}
```

- Negative response to GetPTZPositionList command:

```
{
  "SNAP": {
    "ExecutionStatus": {
      "ErrorText": "Session not found. SessionID=4325",
      "ReturnCode": 200
    }
  }
}
```

## 5.23 GetPTZRoutesList

| Parameter | Description                                   | Default Values |
|-----------|-----------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession | mandatory      |
| EntityID  | Unique user identifier                        | mandatory      |

### Example

- Sample request for getting a list of available routes on camera #38  
<http://192.168.7.221/urlapi/control?Command=GetPTZRoutesList&SessionID=4255&EntityID=38>
- Positive response to the command:
 

```
{
    "SNAP": {
      "PTZRoutesList": {
        "PTZRoutes": {
          "RouteID": 1,
          "Name": "környék",
          "EntityID": 37,

```

```
"Comment": ""
}}}}
```

- Negative response to the command:

```
{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Session not found. SessionID=4325",
  "ReturnCode": 200
}}}
```

## 5.24 GetPTZRouteDefinition

| Parameter | Description                                   | Default Values |
|-----------|-----------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession | mandatory      |
| EntityID  | Unique user identifier                        | mandatory      |
| RouteID   | Unique route identifier                       | mandatory      |

### Example

- Sample request for getting the route definition #1 on camera #38  
<http://192.168.7.221/urLapi/control?Command=GetPTZRouteDefinition&SessionID=4255&EntityID=38&RouteID=1>

- Positive response to the command:

```
{"SNAP": {"PTZRouteDefinition": {"PTZRoute": [
  {
    "PresetPosID": 1,
    "Sequence": 0,
    "RouteID": 1,
    "StayOnTargetTime": 10000,
    "MoveToTargetTime": 0,
    "EntityID": 37
  },
  {
    "PresetPosID": 2,
    "Sequence": 1,
    "RouteID": 1,
    "StayOnTargetTime": 10000,
    "MoveToTargetTime": 0,
    "EntityID": 37
  }
]}}}
```

- Negative response to the command:

```
{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Session not found. SessionID=4325",
  "ReturnCode": 200
}}}
```

## 5.25 LockPTZResource

| Parameter | Description                                   | Default Values |
|-----------|-----------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession | mandatory      |
| EntityID  | Unique identifier of the camera               | mandatory      |

### Example

- Sample request for locking PTZ resource camera #38  
<http://192.168.7.221/urLapi/control?Command=LockPTZResource&SessionID=4255&EntityID=38>
- Positive response to the command:  

```
{ "SNAP": { "ExecutionStatus": { "ReturnCode": 0 } } }
```
- Negative response to the command:  

```
{ "SNAP": { "ExecutionStatus": {  
  "ErrorText": "Session not found. SessionID=4325",  
  "ReturnCode": 200  
} } }
```

## 5.26 RefreshPTZResource

| Parameter | Description                                   | Default Values |
|-----------|-----------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession | mandatory      |
| EntityID  | Unique identifier of the camera               | mandatory      |

### Example

- Sample request for refreshing the lock on the PTZ resource of the camera 38  
<http://192.168.7.221/urLapi/control?Command=RefreshPTZResource&SessionID=4255&EntityID=38>
- Positive response to the command:  

```
{ "SNAP": { "ExecutionStatus": { "ReturnCode": 0 } } }
```
- Negative response to the command:  

```
{ "SNAP": { "ExecutionStatus": {  
  "ErrorText": "Session not found. SessionID=4325",  
  "ReturnCode": 200  
} } }
```

## 5.27 ReleasePTZResource

| Parameter | Description                                   | Default Values |
|-----------|-----------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession | mandatory      |
| EntityID  | Unique identifier of the camera               | mandatory      |

## Example

- Sample request for unlocking the PTZ resource of the camera #38  
<http://192.168.7.221/urLapi/control?Command=ReleasePTZResource&SessionID=4255&EntityID=38>
- Positive response to the command:  

```
{"SNAP": {"ExecutionStatus": {"ReturnCode": 0}}}
```
- Negative response to the command:  

```
{"SNAP": {"ExecutionStatus": {"ErrorText": "Session not found. SessionID=4325", "ReturnCode": 200}}}
```

## 5.28 StartPTZRoute

| Parameter | Description                                   | Default Values |
|-----------|-----------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession | mandatory      |
| EntityID  | Unique identifier of the camera               | mandatory      |
| RouteID   | Unique identifier of the route                | mandatory      |

## Example

- Sample request for starting the PTZ route #1 of the camera #38  
<http://192.168.7.221/urLapi/control?Command=StartPTZRoute&SessionID=4255&EntityID=38&RouteID=1>
- Positive response to the command:  

```
{"SNAP":{"ExecutionStatus":{"ReturnCode":0}}}
```
- Negative response to the command:  

```
{"SNAP": {"ExecutionStatus": {"ErrorText": "Session not found. SessionID=4325", "ReturnCode": 200}}}
```

## 5.29 StopPTZRoute

| Parameter | Description                                   | Default Values |
|-----------|-----------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession | mandatory      |
| EntityID  | Unique identifier of the camera               | mandatory      |

## Example

- Sample request for stopping the PTZ route #1 of the camera #38  
<http://192.168.7.221/urLapi/control?Command=StopPTZRoute&SessionID=4255&EntityID=38>
- Positive response to the command:

```
{"SNAP":{"ExecutionStatus":{"ReturnCode":0}}}
```

- Negative response to the command:

```
{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Session not found. SessionID=4325",
  "ReturnCode": 200
}}}
```

### 5.30 SetPTZPosition

| Parameter  | Description                                   | Default Values |
|------------|-----------------------------------------------|----------------|
| SessionID  | The session identifier given from OpenSession | mandatory      |
| EntityID   | Unique identifier of the camera               | mandatory      |
| PositionID | Identifier of the position                    | mandatory      |

#### Example

- Sample request for directing the PTZ to position #1 of the camera #38  
<http://192.168.7.221/urLapi/control?Command=SetPTZPosition&SessionID=4255&EntityID=38&PositionID=1>

- Positive response to the command:

```
{"SNAP":{"ExecutionStatus":{"ReturnCode":0}}}
```

- Negative response to the command:

```
{"SNAP": {"ExecutionStatus": {
  "ErrorText": "Session not found. SessionID=4325",
  "ReturnCode": 200
}}}
```

### 5.31 PTZCenterClick

| Parameter   | Description                                       | Default Values |
|-------------|---------------------------------------------------|----------------|
| SessionID   | The session identifier given from OpenSession     | mandatory      |
| EntityID    | Unique identifier of the camera                   | mandatory      |
| X           | Horizontal position of the click within the frame | mandatory      |
| Y           | Vertical position of the click within the frame   | mandatory      |
| ImageWidth  | Width of the frame in which X is measured         | mandatory      |
| ImageHeight | Height of the frame in which Y is measured        | mandatory      |

#### Example

- Sample request for moveing the clicked 10, 10 point into the middle of the camera #38  
<http://192.168.7.221/urLapi/control?Command=PTZCenterClick&SessionID=4255&EntityID=38&X=10&Y=10&ImageWidth=640&ImageHeight=480>
- Positive response to the command:

```
{"SNAP":{"ExecutionStatus":{"ReturnCode":0}}}
```

- Negative response to the command:

```
{"SNAP": {"ExecutionStatus": {  
  "ErrorText": "Session not found. SessionID=4325",  
  "ReturnCode": 200  
}}}
```

## 5.32 ContinuousPTZ

| Parameter | Description                                            | Default Values |
|-----------|--------------------------------------------------------|----------------|
| SessionID | The session identifier given from OpenSession          | mandatory      |
| EntityID  | Unique identifier of the camera                        | mandatory      |
| PanSpeed  | Horizontal speed of the PTZ head, -100 >= speed <= 100 | mandatory      |
| TiltSpeed | Vertical speed of the PTZ head, -100 >= speed <= 100   | mandatory      |
| ZoomSpeed | Zoom speed of the PTZ head, -100 >= speed <= 100       | mandatory      |

### Example

- Sample request for continuously moving the PTZ up/left of the camera #38  
<http://192.168.7.221/urLapi/control?Command=ContinuousPTZ&SessionID=4255&EntityID=38&PanSpeed=10&TiltSpeed=10&ZoomSpeed=0>

- Positive response to the command:

```
{"SNAP":{"ExecutionStatus":{"ReturnCode":0}}}
```

- Negative response to the command:

```
{"SNAP": {"ExecutionStatus": {  
  "ErrorText": "Session not found. SessionID=4325",  
  "ReturnCode": 200  
}}}
```

## 5.33 ShowViewOfWindow

| Parameter  | Description                                                                  | Default Values |
|------------|------------------------------------------------------------------------------|----------------|
| SessionID  | The session identifier given from OpenSession                                | mandatory      |
| TargetIP   | IP address of the client on which the action will be executed                | mandatory      |
| TargetUser | The action will be executed on all clients where the given user is logged in | mandatory      |
| WindowID   | Identifier of the Online Monitor window for the action                       | mandatory      |
| PanelName  | Name of the Online Monitor panel which will be                               | mandatory      |

| Parameter | Description | Default Values |
|-----------|-------------|----------------|
|           | activated   |                |

### Example

- Sample request for activating the „view1“ panel on window 1  
[http://192.168.7.221/urLapi/control?Command=ShowViewOfWindow&SessionID=4255&TargetIP=""&TargetUser="admin"&WindowID=1&PanelName="view1"](http://192.168.7.221/urLapi/control?Command=ShowViewOfWindow&SessionID=4255&TargetIP=)
- Positive response to the command:  

```
{ "SNAP": { "ExecutionStatus": { "ReturnCode": 0 } } }
```
- Negative response to the command:  

```
{ "SNAP": { "ExecutionStatus": { "ErrorText": "Session not found. SessionID=4325", "ReturnCode": 200 } } }
```

## 5.34 ShowCameraInViewport

| Parameter   | Description                                                                  | Default Values |
|-------------|------------------------------------------------------------------------------|----------------|
| SessionID   | The session identifier given from OpenSession                                | mandatory      |
| TargetIP    | IP address of the client on which the action will be executed                | mandatory      |
| TargetUser  | The action will be executed on all clients where the givel user is logged in | mandatory      |
| EntityID    | Camera to be displayed                                                       | mandatory      |
| WindowID    | Identifier of the Online Monitor window for the action                       | mandatory      |
| PanelName   | Name of the Online Monitor panel which will be activated                     | mandatory      |
| RowIndex    | Raw in which the camera will be display (or -1 for any position)             | mandatory      |
| ColumnIndex | Column in which the camera will be display (or -1 for any position)          | mandatory      |

### Example

- Sample request for activating camera „1“ in panel „view1“ on window 1  
[http://192.168.7.221/urLapi/control?Command=ShowCameraInViewport&SessionID=4255&TargetIP=""&TargetUser="admin"&WindowID=1&PanelName="view1"&EntityID=1&RowIndex=-1&ColumnIndex=-1](http://192.168.7.221/urLapi/control?Command=ShowCameraInViewport&SessionID=4255&TargetIP=)
- Positive response to the command:  

```
{ "SNAP": { "ExecutionStatus": { "ReturnCode": 0 } } }
```
- Negative response to the command:  

```
{ "SNAP": { "ExecutionStatus": {
```



```
"ErrorText": "Session not found. SessionID=4325",
"ReturnCode": 200
}}}
```

### 5.35 StartSmartGuard

| Parameter     | Description                                                                  | Default Values |
|---------------|------------------------------------------------------------------------------|----------------|
| SessionID     | The session identifier given from OpenSession                                | mandatory      |
| TargetIP      | IP address of the client on which the action will be executed                | mandatory      |
| TargetUser    | The action will be executed on all clients where the given user is logged in | mandatory      |
| ViewGroupName | Name of guard which will be started                                          | mandatory      |

#### Example

- Sample request for starting guard name „G1“ on all clients where the admin user is logged in  
[http://192.168.7.221/urLapi/control?Command=StartSmartGuard&SessionID=4255&TargetIP=""&TargetUser="admin"&ViewGroupName="G1"](http://192.168.7.221/urLapi/control?Command=StartSmartGuard&SessionID=4255&TargetIP=)

- Positive response to the command:  

```
{"SNAP":{"ExecutionStatus":{"ReturnCode":0}}}
```

- Negative response to the command:

```
{"SNAP": {"ExecutionStatus": {
"ErrorText": "Session not found. SessionID=4325",
"ReturnCode": 200
}}}
```

### 5.36 StopSmartGuard

| Parameter  | Description                                                                  | Default Values |
|------------|------------------------------------------------------------------------------|----------------|
| SessionID  | The session identifier given from OpenSession                                | mandatory      |
| TargetIP   | IP address of the client on which the action will be executed                | mandatory      |
| TargetUser | The action will be executed on all clients where the given user is logged in | mandatory      |

#### Example

- Sample request for stopping the guard all clients where the admin user is logged in  
[http://192.168.7.221/urLapi/control?Command=StopSmartGuard&SessionID=4255&TargetIP=""&TargetUser="admin"](http://192.168.7.221/urLapi/control?Command=StopSmartGuard&SessionID=4255&TargetIP=)

- Positive response to the command:  

```
{"SNAP":{"ExecutionStatus":{"ReturnCode":0}}}
```

- Negative response to the command:

```
{"SNAP": {"ExecutionStatus": {"  
  "ErrorText": "Session not found. SessionID=4325",  
  "ReturnCode": 200  
}}}
```