

In-Sight HMI API

API Version	Description
1.0	Released with In-Sight 5.6.0
2.0	Released with In-Sight 5.7.0
2.1	Released with In-Sight 5.7.3. Includes HTTPS and other minor changes.
2.2	Released with In-Sight 5.8.1. Includes Results Queue, Custom View, dialog, findDevices, and other minor changes.
2.3	Released with In-Sight 5.9.2. Custom view format changes, login returns UserAccessInfo, UserAccessInfo includes privileges, setCellName, getLatestResult, loadImage, setEasyView, setHmiPages, and HmiSessionInfo:requestResult
2.4	Released with In-Sight 6.3. Added queryCellResults. Added support for cell ranges in HmiSessionInfo:cellNames. Added CameraInfo::capabilities.
2.5	Release with In-Sight 6.5. Added getCellExpressions.
3.0	Released with In-Sight 22.1.0. Future families of In-Sight are expected to use this API. The root path of requests is now “cam0/hmi”; previous revisions used “system”. Remote graphic requests, HmiSessionInfo:sheetName, beginEdit return value (See Appendix A), and xyCoordinate is used for graphics. Removed support for findDevices. Removed support for Dialogs. Added ffpOnline support. Deprecated and removed support for some fields in HmiSettings. Released with In-Sight 23.1.0: Added support for HmiSessionInfo:autoReady and “autoReady” in CameraInfo:capabilities. Released with In-Sight 23.1.1: Add support for get/setCellCondition, get/set startupOnline and startupJob. Released with 23.2.0: Added LineListDocumentation. Released with In-Sight 23.3.0: Added support for multiple custom views. Added setCustomViewList endpoint to set multiple custom views. Released with In-Sight 24.1.0: Add index parameter to getCustomViewFormat to get any custom view's format that is in the customViewList. Released with In-Sight 24.2.0: Added views into HMIResult. Added View, and ProfileViewCellResult

Document Revision 24.3.0-42

Table of Contents

- 1. Overview..... 5
- 2. Web API..... 5
 - 2.1 Root Resources ("cam0/hmi" or "system")..... 5
 - 2.1.1 Properties..... 5
 - 2.1.2 Methods..... 7
 - 2.1.3 Root Object Types..... 8
 - 2.1.3.1 CameraInfo..... 8
 - 2.1.3.2 State Object..... 9
 - 2.2 Job Resources ("cam0/hmi/job")..... 10
 - 2.2.1 Properties..... 10
 - 2.2.2 Methods..... 10
 - 2.2.3 Job Object Types..... 11
 - 2.2.3.1 EasyViewSettings..... 11
 - 2.2.3.2 EasyViewItem..... 11
 - 2.2.3.3 HmiPages..... 12
 - 2.2.3.4 HmiPage..... 12
 - 2.2.3.5 HmiCustomViewSettings..... 13
 - 2.2.3.6 Sheet Format Object..... 15
 - 2.2.3.6 HmiImageOrientation..... 15
 - 2.3 Settings Resources ("cam0/hmi/settings")..... 16
 - 2.3.1 Properties..... 16
 - 2.3.2 Methods..... 16
 - 2.3.3 Settings Object Types..... 16
 - 2.3.3.1 HmiSettings..... 16
 - 2.3.3.2 UserAccessList..... 18
 - 2.4 HmiSession Resources (e.g. "hs/~1234567890")..... 20
 - 2.4.1 Properties..... 20
 - 2.4.2 Methods..... 21
- 3. Object Types..... 29
 - 3.1 HmiResult..... 29
 - 3.2 HmiSessionInfo..... 31
 - 3.3 View..... 32
 - 3.4 ViewRecord..... 32
 - 3.4.0 ViewPort..... 33
 - 3.4.1 ImageLayer..... 33
 - 3.4.1.1 Image..... 34
 - 3.4.1.2 LinearTransform..... 35
 - 3.4.1.3 ImageLayer for a Processed Image..... 35
 - 3.4.2 GraphicsLayer..... 36
 - 3.4.2.0 Coordinate System..... 36
 - 3.4.2.1 Common Properties of Graphics..... 36

3.4.3 Graphic Types.....	37
3.4.3.1 Annulus.....	37
3.4.3.2 Arc.....	38
3.4.3.3 BeadPath.....	39
3.4.3.4 BlobChain.....	40
3.4.3.5 Circle.....	41
3.4.3.6 ColorMatch.....	42
3.4.3.7 CompositeRegion.....	43
3.4.3.8 Cross.....	45
3.4.3.9 FilledBox.....	46
3.4.3.10 Fixture.....	47
3.4.3.11 Line.....	48
3.4.3.12 LineList.....	49
3.4.3.13 MaskedRegion.....	50
3.4.3.14 Point.....	51
3.4.3.15 Polygon.....	52
3.4.3.16 Polyline.....	53
3.4.3.17 PolylinePath.....	54
3.4.3.18 Rectangle.....	56
3.4.3.19 Region.....	57
3.4.3.20 SubRegion.....	58
3.4.3.21 Text.....	59
3.4.3.22 4Side.....	60
3.5 Cell Results.....	61
3.5.1 FoatResult / HmiFloatResult.....	62
3.5.2 StringResult / HmiStringResult.....	62
3.5.3 EditFloatResult / HmiEditFloatResult.....	62
3.5.4 EditIntResult / HmiEditIntResult.....	63
3.5.5 EditStringResult / HmiEditStringResult.....	63
3.5.6 HmiButtonResult.....	64
3.5.7 CheckBoxResult / HmiCheckBoxResult.....	65
3.5.8 ListBoxResult / HmiListBoxResult.....	65
3.5.9 HmiStatusResult.....	66
3.5.10 MultiStatusResult / HmiMultiStatusResult.....	66
3.5.11 ColorLabelResult / HmiColorLabelResult.....	67
3.5.12 HmiProfileViewResult.....	67
3.5.13 ErrorCellResult / HmiErrorCellResult.....	67
3.5.14 UnsupportedCellResult / HmiUnsupportedCellResult.....	68
3.5.15 DialogResult.....	68
3.5.16 Deprecated Cell Result Types.....	68
3.5.16.1 ButtonResult.....	68
3.5.16.2 EditAnnulusResult.....	69

3.5.16.3 EditPolygonResult..... 70

3.5.16.4 EditCompositeRegionResult..... 71

3.5.16.5 EditMaskedRegionResult..... 72

3.5.16.6 EditPolylinePathResult..... 73

3.5.16.7 EditRegionResult..... 74

3.5.16.8 EditCircleResult..... 74

3.5.16.9 EditLineResult..... 75

3.5.16.10 EditPointResult..... 75

3.5.16.11 StatusResult..... 76

3.5.16.12 StatusLightResult..... 76

3.6 HmiRqState..... 77

3.7 HmiRq..... 78

4. Error Handling..... 79

4.1 HTTP Responses..... 79

Appendix A: Editable Graphic Buttons..... 80

1. Overview

In-Sight cameras (firmware version 5.6+, 22.2.0+) include a WebSocket-based protocol. Clients must use CogSocket (or similar) code to issue requests and handle responses properly. (See the **In-Sight HMI Developers Guide** for more information.) Note: This API is not supported on the D900.

Like HTTP, this protocol uses GET and PUT verbs to access resource values on the camera. The POST operation is used to call a method on the camera. An HTTP POST that has more than one argument, however, must encode them in the url as follows with the 'argsJson' key:

```
POST ...login?argsJson=["admin","mypassword",false,false]
```

This document describes the Web API that is available via that protocol.

2. Web API

In the following section, Properties are retrieved using the GET verb and Methods are called using a POST.

2.1 Root Resources ("cam0/hmi" or "system")

The following items' base resource is "cam0/hmi". So, for example, a request should look like:

GET cam0/hmi/info. *Note: In-Sight devices that implement versions of the API prior to 3.0 (i.e. those before the IS-28XX) have a different root node "system".*

2.1.1 Properties

Resource	Description
availableSessions	Gets the number of free HMI sessions available for connection.
discreteOnline	Gets a boolean value (true or false) that determines whether the discrete I/O online flag is set. This is true, unless discrete I/O has forced the camera offline.
editorAttached	Gets a boolean value (true or false) that designates whether an editor is attached. If true, then the HMI should be in "read-only" mode, not writing to any of the controls. This will be true whenever ISVS, ISE, or VisionView is attached to a camera.
ffpOnline	Gets a boolean value (true or false) that determines whether the FFP online flag is set. This is true, unless the FFP has forced the camera offline. Added with version 3.0 of the API.
info	Gets a CameraInfo object that describes the camera. These are settings or constants that are not job dependent.
isSessionAvailable	Gets a boolean value (true or false) that designates whether a new session may be opened. The HMI settings (set in ISE) designate the maximum number of sessions that may be concurrently active. When no more sessions are available, an error will be returned by a call to openSession . The number of sessions used includes the number of web HMI browsers attached, an ISVS connection, an ISE connection, and any VisionView connections.
job	Gets the job-related information. (See Job Resources below)
jobLoading	Gets a boolean flag that designates whether a job load is in progress.
keepAliveTimeout	Gets or sets the Keep-Alive timeout time in seconds. The default is 30. Any value less than 3 will be coerced to 3. Typically, this value does not need to be changed and is not exposed to the HMI user. The maximum should not be set above 30000.
liveMode	Gets the live mode flag as a boolean value (true or false).
nativeOnline	Gets a boolean value (true or false) that determines whether the native mode online flag is set. This is true unless the native mode has forced the camera offline.
online	Gets the online (run mode)/offline (program mode) state as a boolean value (true or false).

project	This is for Cognex internal use only. Gets the HMI project-related information. The project resources are used by the default web HMI page. A custom HMI does not need to use these resources.
sessions	Gets the number of open sessions. (See isSessionAvailable for information regarding the maximum number of sessions.)
settings	Gets the settings information that is stored on the device. (See Settings Resources below)
softOnline	Gets the software online flag as a boolean value (true or false). This flag should be used in an HMI to change the online/offline state in an HMI (See HmiSession softOnline). For a sensor to go online, besides softOnline , these flags must also be true: nativeOnline , discreteOnline , ffpOnline .
state	Gets the camera's run mode (online/offline) State Object. The individual flags may be retrieved via the resources: online, nativeOnline, discreteOnline, ffpOnline, softOnline, and liveMode.

2.1.2 Methods

Resource	Description
addListener, removeListener	Adds or removes events that notify the client when something occurs. Supported Events: editorAttachedChanged - An event that is raised when the cam0/hmi/editorAttached resource value changed. The payload is the boolean value of that resource. jobChanged - An event that occurs after any of the cam0/hmi/job values have changed. There is no payload for this event. (This was added with version 2.0 of the protocol.) jobLoadingChanged - An event that occurs after the cam0/hmi/jobLoading flag has changed. There payload is the new jobLoading value. (This was added with version 2.0 of the protocol.) jobLoadFailed - An event that occurs after a job load has failed. There payload is an HmiError object. (This was added with version 2.0 of the protocol.) liveModeChanged - An event that occurs when the cam0/hmi/liveMode value changed. The payload for the event is a single boolean value of that resource. sessionDisposed - An event that is raised when the HMI session (previously allocated by cam0/hmi/openSession) has been disposed due to a timeout or some other disconnection. The payload is the session ID string. settingsChanged - An event that occurs after any of the cam0/hmi/settings values have changed. There is no payload for this event. stateChanged - An event that occurs when the cam0/hmi/state value has changed. The payload for the event is 5 boolean values: online, softOnline, nativeOnline, discreteOnline, and ffpOnline. Send { "\$type": "listen", "id": 2, "path": "cam0/hmi/stateChanged" } Got { "\$type":"resp", "id":2 }
openSession	Opens a session and returns a newly created session ID: {SID}. This is a path to the HmiSession resource. Arguments include an optional HmiSessionInfo instance that designates the contents of the results. If the argument is not designated, then all the cells will be included in the results. (NOTE: In this case, the cell names will not be available in the results.) If there are no more connections available, then an error response will be returned. Send { "\$type": "post", "id": 19, "path": "cam0/hmi/openSession", "body": { "\$type": "HmiSessionInfo", "cellNames": ["NamedCell1", "NamedCell2"] } } Success response: { "\$type":"resp", "body":"hs/~1234567890", "id":19 } Error response due to insufficient sessions: { "\$type":"resp", "body":"Access denied: No more HMI connections are available.", "id":15, "error":-1610612728 }

2.1.3 Root Object Types

2.1.3.1 CameraInfo

This object includes information about the camera that doesn't change. Note: some fields, such as the **name**, may change between reboots.

Resource	Description
\$type	"CameraInfo"
acq	Includes properties that designate the resolution of the camera (i.e. "nativeHeight" and "nativeWidth") in pixels. Also "isColor" designates whether the camera is color or greyscale.
capabilities	Version 2.4 of the API adds this field. This array of strings may be used instead of the "supports" properties below. Possible values include "autoReady", "customView", "dialogs", "getCellExpressions", "https", "resultsQueue", and "xyCoordinates". For example: "capabilities":["customView","resultsQueue","xyCoordinates"]
firmwareVersion	The version of the firmware on the camera returned as a string.
hmiProtocolVersion	The version of this API returned as a string.
httpRequestRoot	This optional field can designate a different root to the camera resources when accessed via HTTP. Note that older In-Sight cameras have the root of "sys" when a resource is requested via HTTP.
httpsEnabled	A boolean that designates whether the camera web server is running in HTTPS.
ipAddress	The IP address of the camera.
macID	The MAC Address of the camera.
model	A string that represents the model of the camera.
name	The DNS name of the camera on the network.
serial	The serial number of the camera.
supportsCustomView	A boolean that designates whether the camera supports a custom view. This is deprecated with version 2.4 of the API and capabilities should be used.
supportsDialogs	A boolean that designates whether the camera supports dialogs and wizards. This is deprecated with version 2.4 of the API and capabilities should be used.
supportsHttps	A boolean that designates whether the camera is configured to be able to use HTTPS. This is deprecated with version 2.4 of the API and capabilities should be used.
supportsQueue	A boolean that designates whether the camera supports the results queue. This is deprecated with version 2.4 of the API and capabilities should be used.
usesXYCoordinates	A boolean that designates whether the camera support XY coordinates or just row/column image and graphics coordinates. This is deprecated with version 2.4 of the API and capabilities should be used.

Example:

```
Send { "$type": "get", "id": 3, "path": "cam0/hmi/info" }
Got {
  "$type": "CameraInfo",
  "acq": {
    "$type": "CameraAcqInfo",
    "isColor": false,
    "nativeHeight": 480,
    "nativeWidth": 640
  },
  "firmwareVersion": "6.02.01 (000)", "hmiProtocolVersion": "2.3",
  "httpRequestRoot": "sys", "httpsEnabled": false,
  "ipAddress": "10.10.200.50", "macID": "02-05-85-7F-EB-80",
  "model": "PC-8400", "name": "USRM-0954-1064",
  "serial": "NA", "supportsCustomView": true,
  "supportsHttps": true, "supportsQueue": true,
  "xyCoordinates": false }
```

2.1.3.2 State Object

The state includes the "softOnline", "nativeOnline", "discreteOnline", and "ffpOnline" flags that determine the "online" state of the camera. The "liveMode" setting is also included in this object.

Resource	Description
discreteOnline	A boolean that when true designates that the sensor is not prevented from going online by the discrete input signal. It refers to the Discrete I/O line setting that may be controlled by a PLC or some other device. It is true when the I/O control lines are not used.
ffpOnline	A boolean that when true designates that the sensor is not prevented from going online by the ffp online signal. This is typically true. Added with version 3.0 of the API.
liveMode	A boolean that designates the camera is offline but acquiring images for live mode (i.e. acquisitions are occurring, but the job is not running).
nativeOnline	A boolean that when true designates that the sensor is not prevented from going online by the native mode online signal. This is typically true .
online	A boolean that designates the online state of the camera. A value of true designates that the camera is online (i.e. in run mode).
softOnline	A boolean that when true designates that the sensor is allowed to go online due to the software/editor setting. Note: The is the settings that ISE or the editor used to allow online (i.e. run mode) to be enabled.

Example:

```
Send { "$type": "get", "id": 11, "path": "cam0/hmi/state" }
Got { "$type": "resp", "body": {
  "online": false,
  "softOnline": false,
  "nativeOnline": true,
  "discreteOnline": true,
  "ffpOnline": false,
  "liveMode": false
}, "id": 11 }
```

2.2 Job Resources ("cam0/hmi/job")

The following items' base resource is "cam0/hmi/job". So, for example, a request should look like:
GET cam0/hmi/job/easyView.

2.2.1 Properties

Resource	Description
customViewSettingsList	Gets the list of custom view settings for the job. An array of HmiCustomViewSettings objects are returned or an empty array of HmiCustomViewSettings objects if there are no settings in the job. This was added with version 2.2 of the Protocol.
easyView	Gets the rows that comprise the EasyView. The "items" and "names" may be null, if an EasyView is not defined. An EasyViewSettings object is returned or null if there are no EasyView settings in the job.
jobImageOrientation	Gets the desired orientation of the image as designated in ISE or the job editor. This includes the default rotation, as well as flip (mirroring) operations. An HmiImageOrientation object is returned or null if it is not in the job.
name	Gets the active job name (e.g. "demo.job") running on the sensor (if it is known). Note: there are cases where the firmware does not have an active job name.
pages	Gets a list of pages that are available to the user and accessible through the corresponding View buttons on the HMI. Note that this exists only if there is a job loaded. If no job is loaded, then this property will return null or an empty list. In this case, it is up to the HMI to choose the pages to show. The most logical choice would be a single Image-Only page, but this is totally at the discretion of the HMI.

2.2.2 Methods

Resource	Description
getCustomViewFormat	This method will request the format for the custom view cells in the spreadsheet. An optional boolean argument designates that the dialogs should be included. An index argument that is used to specify which CustomViewFormat in the CustomViewList should be returned. A Sheet Format Object is returned. This was added with version 2.2 of the Protocol.
getSheetFormat	This method will request the format for the cells in the spreadsheet. A Sheet Format Object is returned. This was added with version 2.2 of the Protocol.

2.2.3 Job Object Types

The following objects are stored in a job.

2.2.3.1 EasyViewSettings

The **EasyViewSettings** are stored in the job on an In-Sight camera.

In a spreadsheet job, cells that have symbolic tag names may be added to the EasyView (via the Edit → EasyView Settings menu selection in ISE). In an EasyBuilder job, the Communication app step allows named items to be added to an EasyView.

For each "item", the "caption" is always designated. If the "cellName" is not designated, then the row should be considered a label.

Property	Description
\$type	"EasyViewSettings"
items	An Array of 0 or more EasyViewItem objects that define a row in the EasyView. These items will typically be rendered in order as rows in a table.
names	An array of strings that holds all the names of the EasyViewItem objects in the EasyView.

Example:

```
Send { "$type": "get", "id": 7, "path": "cam0/hmi/job/easyView" }
Got { "$type": "resp", "body": {
  "$type": "EasyViewSettings",
  "items": [{
    "$type": "EasyViewItem",
    "caption": "Blob Minimum Area",
    "cellName": "Blobs_1.Minimum_Area",
    "readOnly": true
  }, {
    "$type": "EasyViewItem",
    "caption": "Blob Region",
    "cellName": "Blobs_1.Composite_Region"
  }
],
  "names": [
    "Blobs_1.Minimum_Area", "Blobs_1.Composite_Region"
  ]
}, "id": 7 }
```

2.2.3.2 EasyViewItem

An **EasyViewItem** is part of the [EasyViewSettings](#). It holds single row of the EasyView.

Property	Description
\$type	"EasyViewItem"
caption	A caption is always designated as the text that should be displayed for the row.
cellName	The cellName identifies the tag in the job that the item is bound to. If no cellName is designated, then the row should be rendered as a label. To obtain the values for the cellName, it should be included in the HmiSessionInfo passed in as argument to cam0/hmi/openSession . The value of the cellName will correspond to the cell result item in the HmiSession's result/cells array that has the same name. See Cell Results for the cell result types that are possible.

readOnly	This boolean flag is false by default and can be used to designate an entry that should not be editable (even though the cellName's type may support editing).
----------	---

2.2.3.3 HmiPages

The **HmiPages** are stored in the job on an In-Sight camera.

They define what pages (or HMI views) should be displayed in the HMI.

Property	Description
\$type	"HmiPages"
items	An Array of 0 or more HmiPage objects that define an HMI page (or HMI View).

Example:

Send { "\$type": "get", "id": 8, "path": "cam0/hmi/job/pages" }

```
Got {"$type":"resp","body": {
  "$type": "HmiPages",
  "items": [
    {
      "$type": "HmiPage",
      "auto": true,
      "name": "EasyViewWithImage"
    },
    {
      "$type": "HmiPage",
      "auto": true,
      "name": "ImageOnly"
    },
    {
      "$type": "HmiPage",
      "auto": true,
      "name": "ImageWithGraphics"
    },
    {
      "$type": "HmiPage",
      "auto": true,
      "name": "EasyViewWithoutImage"
    }
  ]
}, "id": 8 }
```

2.2.3.4 HmiPage

Defines an HMI page (or view).

Property	Description
\$type	"HmiPage"
auto	Deprecated and unused
name	The name that describes the view. (e.g. "ImageOnly")
path	The file/resource containing the page definition. This is false by default and not returned by the device.

2.2.3.5 HmiCustomViewSettings

The **HmiCustomViewSettings** are stored in an array in the job on In-Sight cameras that support the custom view.

Property	Description
top	Top row index of the custom view. (0-599)
bottom	Bottom row index of the custom view. (0-599)
left	Left column index of the custom view. (0-25)
name	Name of the custom view.
right	Right column index of the custom view. (0-25)
x	X coordinate where the custom view should be displayed by default. This is in the pixel space of the full resolution image of the sensor.
y	Y coordinate where the custom view should be displayed by default. This is in the pixel space of the full resolution image of the sensor.
width	The width of the custom view in the pixel space of the full resolution image of the sensor.
height	The height of the custom view in the pixel space of the full resolution image of the sensor.
conditionalCell	This is a string that designates the cell location to use to conditionally update the custom view. The custom view will update when this value is non-zero. This is deprecated with the 3.0 revision of the API; new cameras do not support this.
conditionalUpdate	This is a flag that enables use of the conditionalCell to determine when the custom view will be updated. This is deprecated with the 3.0 revision of the API; new cameras do not support this.
showGraphics	Designates whether the graphics should be displayed for the custom view.
showImage	Designates whether the image should be displayed for the custom view.
showOverlay	Designates whether the spreadsheet should be displayed for the custom view. This is typically true .

Example:

```
[
  {
    "$type": "HmiCustomViewSettings",
    "bottom": 599,
    "height": 240,
    "left": 0,
    "name": "First Custom View",
    "right": 25,
    "showGraphics": true,
    "showImage": true,
    "showOverlay": true,
    "top": 0,
    "width": 320,
    "x": 0,
    "y": 0
  },
  {
    "$type": "HmiCustomViewSettings",
    "bottom": 599,
    "height": 240,
    "left": 0,
    "name": "Second Custom View",
    "right": 25,
    "showGraphics": true,
    "showImage": true,
    "showOverlay": true,
    "top": 0,
    "width": 320,
```

```
    "x": 0,  
    "y": 0  
  }  
]
```

2.2.3.6 Sheet Format Object

The sheet format is stored in the job on an In-Sight camera.

Property	Description
columnWidths	This is an integer array that includes the pixel width for all the columns (26) in a sheet. The value zero designates that the default width should be used for that column.
rowHeights	This is an integer array that includes the pixel height for all the rows (up to 600) in a sheet. The value zero designates that the default height should be used for that row.
cellFormats	This is an array of cell formats. Each cell that has non-default formatting will designate an object that include the following fields (all of which are optional except for the location): • location: The string location of the cell. (e.g. "A0") • fontName: The font name for any text. • fontSize: The font size (in points) for any text. • lineWidth: The line width in pixels for any graphics for the cell. • bold: A Boolean flag to designate a bold font. • italics: A Boolean flag to designate an italics font. • foreColor: The foreground numeric ARGB color for the cell. • backColor: The background numeric ARGB color for the cell. • dp: A number that designates the decimal places of precision that should be used. • hAlign: A number that designates the horizontal alignment of the cell. The values are 0, 1, or 2 to indicate left, center, or right, respectively. • vAlign: A number that designates the vertical alignment of the cell. The values are 0, 1, or 2 to indicate top, middle, or bottom, respectively. Note: The cellFormats property was added in Protocol Version 2.2. Sub-properties hAlign and vAlign were added in Protocol Version 2.3.

2.2.3.6 HmiImageOrientation

The **HmiImageOrientation** is stored with the job and designates how the image should be displayed.

Property	Description
flipHorizontal	This is a boolean that, if true, designates that the image should be rendered flipped horizontally. (Note: this is not used in the default web HMI page.)
flipVertical	This is a boolean that, if true, designates that the image should be rendered flipped vertically. (Note: this is not currently supported)
rotation	Designates the clockwise rotation in degrees. Expected values are 0, 90, 180, 270.

If the flip operations are false, they will be excluded from the JSON string.

Example:

```
Send { "$type": "get", "id": 9, "path": "cam0/hmi/job/jobImageOrientation" }
Got { "$type": "resp", "body": {
  "$type": "HmiImageOrientation",
  "flipHorizontal": true,
  "flipVertical": true,
  "rotation": 90
}, "id": 9 }
```

2.3 Settings Resources ("cam0/hmi/settings")

The settings are items that are not related directly to the job that is running on the device. The settings are stored on the device.

The following items' base resource is "cam0/hmi/settings".

2.3.1 Properties

Resource	Description
skipLogin	Returns a true or false flag that designates whether the HMI should require that the user login. This is currently set to true when there is a default "admin" with no password user in the settings on the camera. In this case, no login is necessary, and the user will have full permissions. Send { "\$type": "get", "id": 30, "path": "cam0/hmi/settings/skipLogin" } Got { "\$type": "resp", "body": true, "id": 30 }
hmi	Gets or sets the HmiSettings that were saved on the sensor. These settings designate how the web HMI server should be configured and what should be available for display on it by default.
userAccessList	Gets the list of users and permissions that are stored on the camera.

2.3.2 Methods

Resource	Description
addListener, removeListener	Adds or removes events that notify the client when something occurs. NOTE: The settings object does not currently expose any events.
save	Saves the current HMI settings to the device. This should be called after they are modified to save them on the device.
load	Loads the HMI settings that are stored on the device. This occurs on startup and usually this method does not need to be called.

2.3.3 Settings Object Types

2.3.3.1 HmiSettings

These settings designate how the web HMI server should be configured and what should be available for display on an HMI by default. These properties designate what controls are available and displayed on the default web HMI page. Note: These do not control the access to the API itself. Access is controlled by which user is logged in.

Resource	Description
\$type	"HmiSettings"
allowAdjustImage	A Boolean that designates whether the operator should be able to adjust the image (e.g. pan/zoom).
allowFilmstrip	A Boolean that designates the filmstrip may be shown if the result queue is enabled.
allowFilmstripSaveImage	A Boolean that designates that an operator should be able to save an image if there are sufficient permissions.
allowFocus	A Boolean that designates that the Focus button should be shown.
allowJobLoad	A Boolean that designates that the Load Job button should be shown.
allowJobSave	A Boolean that designates that the Save Job button should be shown.

allowLocalStorage	A Boolean that designates whether adjustments at the terminal may be saved in the browser's local storage.
allowProcessedImages	A Boolean that designates that processed images should be shown.
allowSoftOnline	A Boolean that designates whether the Online/Offline button should be shown.
allowSwitchView	A Boolean that designates whether the View buttons should be shown.
allowTrigger	A Boolean that designates whether the Trigger button should be shown.
defaultColorScheme	A string that describes the color scheme or theme that should be used on the HMI.
enableHttpImages	When running in HTTPS, this allow images to be transmitted via HTTP to improve performance.
imageResolution	A number that designates the resolution to retrieve the image at. 1=FULL, 2=HALF, 3=QUARTER, 4=EIGHTH
inactivityTimeout	The timeout in seconds after which the logged in user on the HMI should be logged out when not active.
port	The port number that is used for HTTP. NOTE: This field is deprecated with version 3.0 of the API; the default port is 80.
securePort	The port number that is used when HTTPS is active. NOTE: This field is deprecated with version 3.0 of the API; the default secure port is 443.
serverEnabled	A Boolean that designates whether the web server is active and available on the designated port. NOTE: This field is deprecated with version 3.0.
statusStyle	An integer 0 to 2, representing the 3 currently well-defined filmstrip status icon styles used in the default Web HMI page and VisionView (Geometric, OK/NG, and Check/X). The default value is zero, representing the standard pass/fail icons.
useTLS	A Boolean that designates whether TLS should be used when making an HTTPS connection. NOTE: This field is deprecated with version 3.0 of the API.

Example:

```

Send { "$type": "get", "id": 6, "path": "cam0/hmi/settings/hmi" }
Got { "$type": "resp", "body": {
  "$type": "HmiSettings",
  "allowAdjustImage": true,
  "allowFilmstrip": true,
  "allowFilmstripSaveImage": true,
  "allowFocus": true,
  "allowJobLoad": true,
  "allowJobSave": true,
  "allowLocalStorage": true,
  "allowProcessedImages": true,
  "allowSoftOnline": true,
  "allowSwitchView": true,
  "allowTrigger": true,
  "defaultColorScheme": "Default",
  "enableHttpImages": false,
  "imageResolution": 2,
  "inactivityTimeout": 15,
  "statusStyle": 0
}
, "id": 6 }

```

2.3.3.2 UserAccessList

The object holds the list of users and permissions that are stored on the camera. Each user is defined by a **UserAccessInfo** object.

Resource	Description
\$type	"UserAccessInfo"
access	A string that designates the level of permissions granted to the user (e.g. "full", "protected", or "locked"). Note: this is deprecated, and HMIs should use items included in the "privileges" to determine what is available.
accessLevel	An integer designates the level of permissions granted to the user. 0=ADMIN, 1=OPERATOR, 2=MONITOR. Note: this is deprecated, and HMIs should use items included in the "privileges" to determine what is available.
allowOnlineJobSave	A Boolean that grants online job save privileges to this user. Note: this is deprecated, and HMIs should use items included in the "privileges" to determine what is available.
allowStateChange	A Boolean that grants state change privileges to this user. Note: this is deprecated, and HMIs should use items included in the "privileges" to determine what is available.
customView	A Boolean that designates whether the custom view should be shown by default as the view for this user.
ftpRead	A Boolean that grants FTP read access for this user. Note: this is deprecated, and HMIs should use items included in the "privileges" to determine what is available.
ftpWrite	A Boolean that grants FTP write access for this user. Note: this is deprecated, and HMIs should use items included in the "privileges" to determine what is available.
name	The user name.
privileges	An array of strings that designate what privileges are granted to the user. The privileges may only be displayed for the current user for some cameras. In that case, null will be returned. The privileges for each access level are as follows: Admin: ["IS.CFG", "IS.IMAGE", "IS.MNT", "IS.OPS", "IS.FILE", "IS.OPENFILE", "IS.WRITEFILE", "IS.JOB", "IS.OPENJOB", "IS.EDITJOB", "IS.2AUTH", "IS.CFGJOB", "IS.CSTMALL", "IS.SAVE", "IS.SAVEJOBAS", "IS.TESTRUN", "IS.ONLINEOFFLINE"] Operator: ["IS.IMAGE", "IS.OPS", "IS.OPENJOB", "IS.CFGJOB", "IS.CSTMALL", "IS.ONLINEOFFLINE"] Monitor: ["IS.CSTMALL"]

Example:

Send { "\$type": "get", "id": 10, "path": "cam0/hmi/settings/userAccessList" }

Got {"\$type": "resp", "body":

```
[
  {
    "$type": "UserAccessInfo",
    "access": "full",
    "accessLevel": 0, // 0=ADMIN, 1=OPERATOR, 2=MONITOR
    "allowOnlineJobSave": false,
    "allowStateChange": true,
    "customView": true,
    "ftpRead": true,
    "ftpWrite": true,
    "name": "admin",
    "privileges": null
  }
]
```

```

},
{
  "$type": "UserAccessInfo",
  "access": "locked",
  "accessLevel": 2,
  "allowOnlineJobSave": false,
  "allowStateChange": false,
  "customView": false,
  "ftpRead": false,
  "ftpWrite": false,
  "name": "monitor",
  "privileges": null
},
{
  "$type": "UserAccessInfo",
  "access": "protected",
  "accessLevel": 1,
  "allowOnlineJobSave": false,
  "allowStateChange": true,
  "customView": false,
  "ftpRead": true,
  "ftpWrite": false,
  "name": "operator",
  "privileges": null
}
], "id": 10 }

```

Current User Example:

Send { "\$type": "get", "id": 49, "path": "hs/~50684555/currentUser" }

```

Got {
  "$type": "UserAccessInfo",
  "access": "full",
  "accessLevel": 0,
  "allowOnlineJobSave": false,
  "allowStateChange": true,
  "customView": false,
  "ftpRead": true,
  "ftpWrite": true,
  "name": "admin",
  "privileges": [
    "IS.CFG", "IS.IMAGE", "IS.MNT", "IS.OPS", "IS.FILE", "IS.OPENFILE", "IS.WRITEFILE",
    "IS.JOB", "IS.OPENJOB", "IS.EDITJOB", "IS.2AUTH", "IS.CFGJOB", "IS.CSTMALL", "IS.SAVE",
    "IS.SAVEJOBAS", "IS.TESTRUN", "IS.ONLINEOFFLINE"
  ]
}, "id": 49 }

```

2.4 HmiSession Resources (e.g. "hs/~1234567890")

The HMI Session resources are exposed via the url that is returned from **cam0/hmi/openSession**. This is of the form: **cam0/hmi/hs/~1234567890**. So, each resource below will begin with the session ID. For example: GET {SID}/sessionInfo.

2.4.1 Properties

Resource	Description
currentUser	Gets the current user object (UserAccessInfo) that is logged into the session or null if no user is logged in. Note that the privileges field will be populated with specific privileges that are allowed for the user.
id	Gets the id that identifies the session (e.g. "~50684555").
liveMode	Gets or sets the live mode flag as a boolean value (true or false). Setting the value requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.OPS" privilege.
result	Gets the latest results (HmiResult) for the session. The instance will always include an "acqImageView" as a predefined place to get the default view. If a user is not logged in, then this property will return null .
rq	Gets the resource that controls the camera's result queue. See HmiRq .
sessionInfo	Gets the info that designates what is included in the session results. The "cellNames" array designates specific named cells to be included. Returns an HmiSessionInfo object or null if one was not set.
softOnline	Gets or sets the soft online flag as a boolean value (true or false). Setting the value requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.ONLINEOFFLINE" privilege.
startupJob	Gets or sets the startup job name. Setting the value requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.ONLINEOFFLINE" privilege.
startupOnline	Gets or sets the flag (i.e. true or false) that designates whether the camera should start up online. Setting the value requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.ONLINEOFFLINE" privilege.

2.4.2 Methods

Resource	Description
addListener, removeListener	Adds or removes events that notify the client when something occurs. "{SID}" events include: resultChanged - An event that provides the latest results. This event will only be raised when the session is ready for results and there is a new result available. Send { "\$type": "listen", "id": 40, "path": "hs/~50684555/resultChanged" } Got { "\$type": "resp", "id": 40 }
backupDialogData	This method backs up the dialog area of the sheet. This is used in conjunction with freeDialogData to allow any of the changes to the dialog cells to be cancelled (i.e. cell value changes reverted to the original values). This method requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.CFGJOB" privilege. NOTE: This method is not supported in version 3.0 of the API.
beginEdit	Returns a ViewRecord or an "Editing" object for a designated cell's graphics. The optional argument designates a cell that is the source of the graphic that will be edited. If no argument is specified, then just the acquired image will be in the record. This method is typically used before editing a graphic, where it is desirable to show a fixed view for editing. When the editing operation is complete, the HMI should call editEdit to free the associated resources on the camera. New devices (that support at least revision 3.0 of the API) will return an "Editing" object (e.g. EditingRegion). When an "Editing" object is available for editing a graphic, then it should be used to determine the edit flags (e.g. "rotate", "move", "bend", etc) for the graphic. In this case, the graphic flags will not be included in the CellResult for the designated cell to be edited. A user must be logged in to call this method.
closeDialog	This method closes the dialog that is currently open. This method requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.CFGJOB" privilege. NOTE: This method is not supported in version 3.0 of the API.
createNewJob	Creates a new, empty job. This method requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.OPENJOB" privilege.
dispose	Disposes the session. Any resources held by the session on the device will be released (and likely freed). In particular, any results that were received by a client should not be used to request images as after dispose is called, as there is no guarantee that they will still be published by the device. Send { "\$type": "post", "id": 48, "path": "hs/~50684555/dispose" } Got { "\$type": "resp", "id": 48 }

endEdit	Frees the resources associated with the editing view by a previous call to beginEdit. This method is typically called when the editing operation for a graphic is complete. Send { "\$type": "post", "id": 46, "path": "hs/~50684555/endEdit" } Got { "\$type": "resp", "body": 0, "id": 46 }
findDevices	This method returns an array of devices discovered on the network. It uses the CogNamer client protocol (UDP) to find the devices on the subnet. This method will take at least 3 seconds to complete. It will return at most 200 devices. Each returned object in the array has a name, ipAddress, and model. There will also be a field for each service (i.e. port for various services) provided by the device. A user must be logged in to call this method. NOTE: This method is not supported in version 3.0 of the API. Send { "\$type": "post", "id": 6, "path": "hs/~6c212f11/findDevices" } Got { "\$type": "resp", "body": [{ "model": "PC-8400", "ftp": 21, "ipAddress": "10.12.220.48", "hmi-http": 8087, "telnet": 23, "name": "USRM-0954-1064", "http": 8087 }], "id": 6 }
freeDialogData	Frees the cell values that were saved with the call to backupDialogData. The Boolean argument to this method should be set to true to restore the values of the cells to the backup values. This method requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.CFGJOB" privilege. NOTE: This method is not supported in version 3.0 of the API.
getAllCellNames	Gets all of the cell names in the job as a dictionary of cell locations and names. Send { "\$type": "post", "id": 47, "path": "hs/~50684555/getAllCellNames" } Got { "\$type": "resp", "body": { "A3": "Acquisition.Trigger", "A9": "Acquisition.Exposure_Region", "B21": "Blobs_1.Pass", "B23": "Blobs_1.Blob_Count" }, "id": 47 }
getCellCondition	Gets the cell state condition of a cell. The arguments are the cell name/location and the string condition for that cell. A user will need to be logged in to get the cell state.
getCellExpression	Gets the expression of a cell. The argument is the cell name/location of a cell. If the cell is protected, it will not be able to be read. A user will need to be logged in to get the expressions. Send { "\$type": "post", "id": 13, "path": "hs/~5fec2937/getCellExpression", "body": "MyEditInt" } Got { "\$type": "resp", body: "EditInt(0,255)", "id": 13 }

getCellExpressions	Gets the expressions for a range of cells and optionally allows a Boolean argument to include a data ID for the cells that have extra data beyond the expression. The data ID might be a CRC, checksum, or other data identifying number. This method may be used to track changes to a job. Cameras that support this capability will include “getCellExpressions” in the “capabilities” of the CameraInfo. This was first added to the 6.5 and 22.2.1 firmware. <pre> Send { "\$type": "post", "id": 13, "path": "hs/~5fec2937/getCellExpressions", "body": ["A0:B10", true] } Got { "\$type": "resp", body:" [{ "A0": { "expr": "AcquireImage()" }, "A2": { "expr": "FindPatMaxRedLine(\$A\$0,0,0,0,100,80,440,320,0,0,0,0,1,50,0,1,0,0,- 15,15,100,100,70,0,60,5000,0)" }, "A4": { "dataId": 276236069, "expr": "TrainPatMaxRedLine(\$A\$0,0,0,0,100,80,440,320,0,0,0,0,1,1,4,1,20,0,0,0,0)" }, "B1": { "expr": "'Index'" }, "B2": { "expr": "0" }, "B4": { "expr": "GetTrained(A4)" } }]","id":13} </pre>
getLatestResult	Gets the latest result by updating the result property, and returning an HmiResult. This was added with Version 2.3 of the protocol.
getSessionIDs	Gets the open sessions IDs. This method requires Full ("admin") access, or specifically, the "IS.CFG" privilege. <pre> Send { "\$type": "post", "id": 38, "path": "hs/~50684555/getSessionIDs" } Got { "\$type": "resp", "body": ["hs/~50684555", "hs/~2345678901"], "id": 38 } </pre>
getSheetResult	For internal use only. Gets the SheetResult that is held by the HmiResult. A user must be logged in to call this method.
keepAlive	Keep alive message to detect that the HMI (client/browser) is still functioning and not locked up or shut down. When the keepAliveTimeout time has expired without receiving one of these, it will dispose of all resources associated with the session and its connection. When this occurs, the cam0/hmi/sessionDisposed event will be raised. <pre> Send { "\$type": "post", "id": 45, "path": "hs/~50684555/keepAlive" } Got { "\$type": "resp", "id": 45 } </pre>

listFiles	Returns an array of FileListItem objects that represent the files in a directory. The directory can either be on the sensor itself or on a remote FTP server. The optional arguments for this method are: the directory or URL that designates the directory on an ftp server, a base 64 encoded FTP user name, and a base 64 encoded FTP password. The user must have FTP read permissions to use this method. This method was added for version 2.0 or the protocol. Besides the name of the file, the fields of the FileListItem include the attribute (1: file, 8: directory) and the size measured in bytes. <pre>Send { "\$type": "post", "id": 33, "path": "hs/~50684555/listFiles" } Got { "\$type":"resp", "body": [{ "\$type": "FileListItem", "attributes": 1, "name": "MyNewJob.job", "size": 321130 }, { "\$type": "FileListItem", "attributes": 8, "name": "SDCARD", "size": 0 }, ...], "id":33 }</pre>
loadImage	This method loads an image from the base 64 encoded ASCII data for an image. The second parameter is an optional Boolean flag that allows the JPG format to be designated (rather than the default BMP format). The third parameter is also optional and allow an image name to be designated for devices that support it. This method requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.IMAGE" privilege. This was added with Version 2.3 of the protocol.
loadJob	This method loads the designated job file on the sensor as its active job. The first argument designates the file name or address of the remote FTP device. The optional second argument is a base 64 encoded FTP username. The optional third argument is a base 64 encoded FTP password. Loading a job requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.OPENJOB" privilege. <pre>Send { "\$type": "post", "id": 33, "path": "hs/~50684555/loadJob", "body": "MyJob.job" } Got { "\$type":"resp", "body":0, "id":33 }</pre>

loadJobData	This method loads the designated job data as the current job. The argument must be an HmiNamedContent object. The content must be data that is either formatted as a base 64 encoded ASCII string or a dictionary with a "base64" keyed item that holds the data. NOTE: This method should be called via an HTTP request, and not using CogSocket. Sending this data via HTTP is more efficient and the web socket connect will not handle large incoming data due to web socket data size limitations. Loading a job requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.OPENJOB" privilege. <pre>//At this point, the FileReader has read the content of the file. var uri = reader.result; var content = uri.replace(/data:(.*)"base64/,/, ''); let hmiNamedContent = { "\$type": "HmiNamedContent", "name": file.name, "content": content }; loadJobData(hmiNamedContent, (resp) => { if (resp == null) { alert("Error loading job"); } }); ... public loadJobData(hmiNamedContent: any, onResponse: (resp: any) => void) { // POST the request to the sensor, because a web socket request will not allow the loading // of a large file. let urlBase: string = "http://" + this.inspectionAddress; let requestUrl = urlBase + "/sys/" + this._sessionId + "/loadJobData"; httpPostAsync(requestUrl, hmiNamedContent, (resp) => { onResponse(resp); }); }</pre>
login	Logs in as a user for the session. The username and password must be designated as arguments. The username and password should be 64 bit ASCII encoded before transmission. There is an optional third parameter (added with version 2.3 of the protocol), a boolean, that may be used to designate whether to encode the arguments. Set this third argument to false when the arguments will not be encoded. There is an optional fourth parameter (added with version 2.3 of the protocol), a boolean, that may be used to designate to return the UserAccessInfo object instead of the string access level. Returns a string that represents the access level a response that includes an error. <pre>Send { "\$type": "post", "id": 4, "path": "hs/~50684555/login", "body": ["YWRtaW4=", ""] } Success response: { "\$type": "resp", "body": "full", "id": 4 } Error response due to login failure: { "\$type": "resp", "body": "Login failed", "id": 20, "error": -1610612728 }</pre>
logoff	Logs the user out of the session. The session is not closed, however, and login can be used again. After the call, the session resources that are secured will return an Access Denied response if access is attempted. <pre>Send { "\$type": "post", "id": 49, "path": "hs/~50684555/logoff" }</pre>

manualTrigger	Triggers an acquisition. Manual triggering requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.OPS" privilege. Send { "\$type": "post", "id": 43, "path": "hs/~50684555/manualTrigger" } Got { "\$type": "resp", "id": 43 }
openDialog	This method opens a dialog specified at the designated Dialog cell location (or name). This method requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.CFGJOB" privilege. It also cannot be called when a job is loading. NOTE: This method is not supported in version 3.0 of the API.
queryCellResults	This method takes an array of cell range strings or cell names and returns an array of HmiCellResults for the result held by the session. This was added with version 2.4 of the protocol. For example: <pre>hs/~033b6e6c/queryCellResults [["A2:A7", "BlobCell"]] Result: [{ "\$type": "StringResult", "error": false, "location": "A2", "name": "BlobCell", "data": "\u0007Blobs" }, { "\$type": "StringResult", "location": "A7", "name": "A7", "data": "test" }]</pre>
ready	This is a method that updates the result for the session when one is available from the device. A resultChanged event is raised when the result resource changed. Send { "\$type": "post", "id": 33, "path": "hs/~50684555/ready" } Got { "\$type": "resp", "body": 0, "id": 33 }
saveJob	This method saves the current job as the designated job. The first argument designates the file name or address of the remote FTP device with the filename included. The optional second argument is a base 64 encoded FTP username. The optional third argument is a base 64 encoded FTP password. Saving a job requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.SAVEALL", "IS.SAVE", or "IS.SAVEJOBAS" privilege. Send { "\$type": "post", "id": 33, "path": "hs/~50684555/saveJob", "body": "MyJob.job" } Got { "\$type": "resp", "body": 0, "id": 33 }

saveJobData	This method requests the job as a base 64 encoded ASCII array. The client may then save this locally to a file (after converting it from base64 to a regular ASCII file). Saving a job requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.SAVEALL", "IS.SAVE", or "IS.SAVEJOBAS" privilege. <pre>Send { "\$type": "post", "id": 17, "path": "hs/~3da91e3c/saveJobData", "body": "MyJob.job" } Got { "\$type":"resp", "body":{"\$type":"Byte[]", "base64":"J1ZF...NCg=="}, "id":17 }</pre>
setCellCondition	Sets the cell state condition of a cell. The arguments are the cell name/location and the string condition for that cell. This requires Full ("admin") access, or specifically, the "IS.EDITJOB" privilege.
setCellExpression	Sets the expression of a cell. The arguments are the cell name/location and the string expression for that cell. This requires Full ("admin") access, or specifically, the "IS.EDITJOB" privilege. If the cell is protected, it will not be able to be written. <pre>Send { "\$type": "post", "id": 13, "path": "hs/~5fec2937/setCellExpression", "body": ["MyEditInt", "EditInt(0,255)"] } Got { "\$type":"resp", "id":13 }</pre>
setCellName	Sets the name of a cell. This defines a symbolic name that may be used to identify the cell. The arguments are the cell location and the name for that cell. This requires Full ("admin") access, or specifically, the "IS.EDITJOB" privilege.
setCellValue	Sets the value of a cell to an object/value. The arguments are the cell name/location and the value to send to that cell. Note that the value object may be a complex type. So, for an EditRegion cell, it will be a "Region" instance. See Appendix A: Editable Graphic Buttons for the data values that can be used from the corresponding Edit Graphic buttons. This method requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.CFGJOB" privilege. <pre>Send { "\$type": "post", "id": 13, "path": "hs/~5fec2937/setCellValue", "body": ["MyEditInt", 1] } Got { "\$type":"resp", "id":13 }</pre>
setCellValues	Wide setter for cell values. Argument is a dictionary of cell locations and values. This method requires Full ("admin") or Protected ("operator") access, or specifically, the "IS.CFGJOB" privilege. <pre>Send { "\$type": "post", "id": 13, "path": "hs/~5fec2937/setCellValues", "body": [{ "MyEditInt": 1, "MyEditString": "test" }] } Got { "\$type":"resp", "id":13 }</pre>

setEasyView	This method sets the EasyView for a job. The argument is an array of EasyViewItems. This method requires Full ("admin") access, or specifically, the "IS.CFG" privilege. This was added with Version 2.3 of the protocol.
setHmiPages	This method sets the HMI pages for a job. The argument is an array of HmiPage items. This method requires Full ("admin") access, or specifically, the "IS.CFG" privilege. This was added with Version 2.3 of the protocol.
setSessionInfo	Sets the info that designates what is included in the session results. An HmiSessionInfo instance must be provided as an argument. This method will return the latest result (i.e. HmiResult) generated using the designated session info. See HmiSessionInfo.

3. Object Types

3.1 HmiResult

The **HmiResult** object is returned as the "result" of the session. It is the latest result since "ready" was called for that session. The fields are as follows:

Property	Description
\$type	"HmiResult"
id	An integer id that uniquely identifies the result for the session. As each result is generated for a session, the id will be incremented.
acqImageView	ISE: The primary ViewRecord for the camera acquisition that is produced by AcquireImage. ISVS: The primary ViewRecord for the camera acquisition that is produced by AcquireImage, InputImage, or GetGreyscale.
acqPointCloudView	The primary ViewRecord for the camera acquisition that is produced by InputPointCloud.
cellTagVer	An integer that identifies the version of the cell results.
cells	The array of cell results that are included in the results for the session.
jobStatus	The jobStatus is an integer that describes the result, spreadsheet or EasyBuilder, of the job. It is one of the following: 0 (None), 1 (Pass), 2 (Fail), 3 (Warn)
jobTagVer	An integer that identifies the version of the job tag. (This is used primarily for diagnostic purposes to identify when the items like the EasyView table are changed in the job.)
logicVer	An integer that identifies the version of the cells in the job. (This is used primarily for diagnostic purposes to identify when the cells in the spreadsheet have changed.)
queuedResult	A Boolean flag that designates whether the result is from the queue. The default value is false.
rq	This object holds the state of the result queue when the result was generated. The default value for this object is null , returned when the queue is not enabled on the camera. See HmiRqState for additional information.
views	A dictionary of all the Views produced by the execution of the job. The keys are the names of the views and the value are a View object that contain all the view information.

Example:

```
Send { "$type": "get", "id": 36, "path": "hs/~50684555/result" }
Got { "$type": "resp", "body": {
  "$type": "HmiResult",
  "acqImageView": {
    "$type": "ViewRecord",
    //... refer to the ViewRecord section to see full specification
  },
  "acqPointCloudView": {
    "$type": "ViewRecord",
    //... refer to the ViewRecord section to see full specification
  }
},
"cellTagVer": 478,
"cells": [
  { "$type": "FloatResult", "location": "S24", "name": "PlotData", "data": 7 },
  { "$type": "StringResult", "location": "T13", "name": "PlotCell", "data": "#Plot" }
],
"id": 2,
"jobStatus": 0,
"jobTagVer": 5,
"logicVer": 20,
"views": {
  "A0View": {
    "$type": "View",
```

```
    //... refer to the View section to see full specification
  },
  "A1View": {
    "$type": "View",
    //... refer to the View section to see full specification
  }
},
"id":36 }
```

3.2 HmiSessionInfo

The **HmiSessionInfo** designates what is included in the session results.

Property	Description
\$type	"HmiSessionInfo"
autoReady	Designates a session that does not require a 'ready' message to be sent to request the next result. NOTE: This option should be used with caution and is not typically set. Only a single session should be enabled with this option as it is provided for archiving results and not a throttled HMI connection. When this option is enabled, it is recommended that the result queue is enabled on the camera allowing it to queue all results. Otherwise, requests for images and remote graphics may fail. Enabling the queue will increase the time that the images and graphics may be requested because the camera will only hold the last image processed or any that are currently in the queue. This was added with the 23.1.0 firmware, and the "autoReady" string is included in the CameraInfo capabilities when it is supported.
cellNames	An array of cell names that designates what cell results to include in the HmiResult provided by the HmiSession. In version 2.4 of the protocol, a range of cells may also be designated in the array. For example, adding "A0:Z599" would select all cells in the sheet.
enableQueuedResults	A Boolean flag that designates whether the queued results should be provided by the HmiSession when the camera result queue is frozen. This is false by default.
ignoreEditorAttached	A Boolean flag that allows the API calls to be permitted when an editor is attached. This is false by default. This was added with version 3.0 of the protocol.
includeCustomView	A Boolean flag that designates whether to include the custom view range of cell in the cell results in the HmiResult provided by the HmiSession .
requestResult	A Boolean flag that designates results will be requested, and the payload of the resultChanged event should be empty. This was added with version 2.3 of the protocol.
sheetName	A string that designates the name of the sheet for which results will be returned. This was added with version 3.0 of the protocol for devices that may support multiple sheets.

Example:

```
Send { "$type": "get", "id": 39, "path": "hs/~50684555/sessionInfo" }
Got { "$type": "resp", "body": {
  "$type": "HmiSessionInfo",
  "cellNames": ["NamedCell1", "NamedCell2"]
}, "id": 39 }
```

3.3 View

The **View** holds the title, the name of the cell that produces this view and the **ViewRecord** associated with this **View**.

Property	Description
\$type	"View"
record	The ViewRecord associated with this view.
source	Reference to the name of the cell that produces this view.
title	The title of the view. This is the cell's name or the cell's location referenced in the source with the suffix "View". E.g.: the view of the cell A0 will have the title A0View. The view of the named cell "MyImage" will have the title MyImageView.
url	A URL that identifies the View . An HMI will typically not need to use this URL.

Example:

```
Send { "$type": "get", "id": 34, "path": "hs/~50684555/result/views/A0View" }
Got { "$type": "resp", "body": {
  "$type": "View",
  "record": {
    "$type": "ViewRecord",
    "id": 10000000000002,
    "url": "/cam0/app/views/001000000000002",
    "heightColor": null,
    "layers": [
      {
        "$type": "ImageLayer",
        "url": "/cam0/img/001000000000004",
        "height": 500,
        "image": {
          "$type": "Image",
          "id": 10000000000004,
          "url": "/cam0/img/001000000000004",
          "frozen": true, "height": 500, "mask": null,
          "offsetX": 0, "offsetY": 0, "staticTransform": null,
          "transform": {
            "$type": "LinearTransform",
            "m00": 0,142857149243355, "m01": 0, "m10": 0, "m11": -0,142857149243355, "tx": -50,0714285746217,
            "ty": 71,3571450337767
          },
          "width": 700,
          "acquisitionInfo": null,
          "bitsPerPixel": 8,
          "imageFormat": 8,
          "isColor": false,
          "name": "",
          "orientation": 0,
          "readOnly": true
        },
        "mask": null,
        "staticTransform": null,
        "transform": null,
        "width": 700
      },
      { "$type": "SVGLayer", "url": "/cam0/app/views/001000000000002/layers/1/svg", "contentType": 0,
        "transform": null },
      { "$type": "GraphicsLayer", "url": "/cam0/app/views/001000000000002/layers/2/graphics" }
    ],
    "profileRectangle": null,
    "source": "Inspection",
    "viewport": { "$type": "Viewport", "height": 500, "width": 700 }
  },
  "source": "A1",
  "title": "A1View",
  "url": "/cam0/app/views/001000000000002"
}
```

3.4 ViewRecord

The **ViewRecord** holds the image and graphics that are associated with the **HmiResult**.

Property	Description
\$type	"ViewRecord"
bounds	The bounds of the view in camera pixel coordinates. This is only designated when a view is constructed for editing a graphics. Otherwise, it is null . When it is provided, it will only be a Rectangle object with "x", "y", "h", "w" properties. (See beginEdit)
layers	An array that contains ImageLayer and GraphicsLayer instances. Typically, this array will include an ImageLayer followed by a GraphicsLayer . If there are processed images and graphics in the results, then there will be additional layers. Layers should be rendered in order.
source	A string that describes the source of the view. Currently, this is always "sheet".
url	A URL that identifies the ViewRecord . An HMI will typically not need to use this URL.
viewport	The Viewport designates the location and size of the area (in camera pixel coordinates) that the layers are mapped to.

Example:

```
Send { "$type": "post", "id": 34, "path": "hs/~50684555/beginEdit" }
Got {"$type":"resp","body": {
  "$type": "ViewRecord",
  "bounds": { "$type": "Rectangle", "x": 0, "y": 0, "h": 480, "w": 640 },
  "layers": [
    {
      "$type": "ImageLayer",
      "url": "/sys/img/001000000000008",
      "height": 480,
      "image": {
        "$type": "Image",
        "acquired": true, "bitsPerPixel": 8, "frozen": false,
        "high": 480, "offsetX": 0, "offsetY": 0, "sequenceNumber": 2,
        "url": "/sys/img/001000000000008",
        "wide": 640
      },
      "transform": null,
      "width": 640
    }
  ],
  "source": "Sheet",
  "viewport": { "$type": "Viewport", "height": 480, "width": 640 }
}, "id": 34 }
```

3.4.0 ViewPort

The **Viewport** holds the location and the size (in camera pixel coordinates) that the layers should be mapped to.

Property	Description
\$type	"Viewport"
height	The total height in pixels to render the ViewRecord .
width	The total width in pixels to render the ViewRecord

3.4.1 ImageLayer

The **ImageLayer** holds the information necessary to retrieve and render an image. An image should be retrieved by making an HTTP request via its url.

Property	Description
\$type	"ImageLayer"

height	The total height in pixels of the image buffer on the camera.
image	The Image object that provides additional information about the image.
mask	The mask provides the Region for the valid pixels that are contained in the Image. For the main image this will typically be null . For a processed image, the actual image acquired and saved by the camera may include additional pixels. Some In-Sight cameras acquire on a 4-byte boundary, so images may include additional pixels. The mask region provides the bounds of the valid pixels that are included.
transform	The transform that is used to map the image to the ViewPort . For the main acquired image, this will typically be null because the image maps directly to the ViewPort . For a processed image, this will be a LinearTransform .
url	The URL that provides the path to retrieve the image via HTTP.
width	The total width in pixels of the image buffer on the camera.

3.4.1.1 Image

The **Image** object is included in an **ImageLayer**.

Property	Description
\$type	"Image"
acquired	A flag ("true" or "false") that designates whether the image is from an acquisition or was retrieved from storage.
bitsPerPixel	The bits per pixel that comprise the image.
col0	The column of the first pixel.
frozen	A Boolean flag that is true if the image was read from a file or storage.
height	The height in pixels of the image.
high	The height in pixels of the image. (This is the same as "high")
offset	The X offset of the first pixel. (On an In-Sight camera, this is the same as "row0")
offset	The Y offset of the first pixel. (On an In-Sight camera, this is the same as "col0")
row0	The row of the first pixel.
sequenceNumber	The number that uniquely identifies the image.
url	The URL that provides the path to retrieve the image via HTTP.
wide	The width in pixels of the image.
width	The width in pixels of the image. (This is the same as "wide")

3.4.1.2 LinearTransform

The **LinearTransform** provides an offset into the parent image for rendering the associated Image.

Property	Description
\$type	"LinearTransform"
tx	The X offset in pixels. See 3.4.2.0 Coordinate System .
ty	The Y offset in pixels. See 3.4.2.0 Coordinate System .

Example:

```
{
  "$type": "LinearTransform",
  "tx": 240,
  "ty": 100
}
```

3.4.1.3 ImageLayer for a Processed Image

The following example is an object that is returned for a processed image. Note that it includes the additional “mask” and “transform” fields.

Example:

```
{
  "$type": "ImageLayer",
  "url": "/sys/img/000000000000011",
  "height": 188,
  "image": {
    "$type": "Image",
    "acquired": false,
    "bitsPerPixel": 8,
    "col0": 100,
    "frozen": true,
    "height": 188, "high": 188,
    "offsetX": 240, "offsetY": 100,
    "row0": 240,
    "sequenceNumber": 0,
    "url": "/sys/img/000000000000011",
    "wide": 152, "width": 152
  },
  "mask": {
    "$type": "Region",
    "color": -16777077,
    "angle": 0,
    "curve": 0,
    "h": 188, "w": 149, "x": 240, "y": 102
  },
  "transform": {
    "$type": "LinearTransform",
    "tx": 240, "ty": 100
  },
  "width": 152
}
```

3.4.2 GraphicsLayer

The **GraphicsLayer** holds the graphics that should be rendered over any previous image or graphics layers.

Property	Description
\$type	"GraphicsLayer"
graphics	An array of zero or more graphics objects (e.g. "Region" object, "Circle" object, etc.) as defined below. Note: This will not be returned if a url is designated.
url	If a url path is designated, then the graphics must be requested as a remote resource.

3.4.2.0 Coordinate System

New In-Sight cameras (IS-28XX and after that implement at least revision 3.0 of the API) may return **true** for **info/xyCoordinates**. This designates that the sensor supports a standard XY coordinate system and not row/column (i.e. X-vertical, y-horizontal) coordinates. In this case the, the rotation for the graphics will also be in the clockwise direction.

Older In-Sight cameras have a coordinate system has the origin in the top, left corner. The Y-axis is horizontal and increasing to the right. The X-axis is vertical and increasing downward. When an angle is specified, its units will be degrees and the positive direction will be counterclockwise.

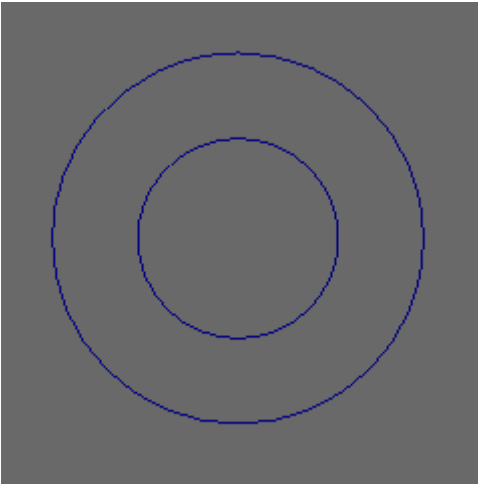
3.4.2.1 Common Properties of Graphics

Each graphic object that may be part of the "graphics" array in the **GraphicsLayer** will have these common properties:

Property	Default	Description
\$type		The specific type of the graphic.
color		The ARGB color of the graphic as a number. TypeScript Example to convert to a color string: <code>let color: string = '#' + (Number)(color).toString(16).substr(2);</code>
lineThickness	1	The line thickness to use to render the graphic.
source		The location of the cell that generated the graphic.

3.4.3 Graphic Types

3.4.3.1 Annulus



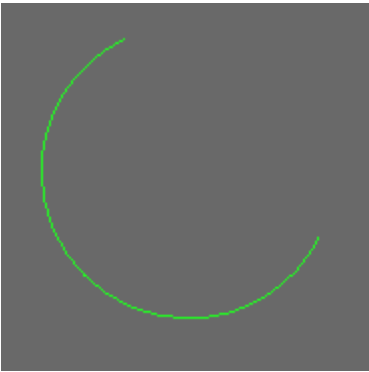
The “EditAnnulus” function will produce a “Annulus” graphic.

Property Name	Type	Default Value	Description
\$type	string	"Annulus"	Annulus class name.
innerRadius	float		Radius of the inner circle of the annulus.
outerRadius	float		Radius of the outer circle of the annulus.
x	float		X co-ordinate of the center of the annulus.
y	float		Y co-ordinate of the center of the annulus.

Example:

```
{
  "$type": "Annulus",
  "color": -16777088,
  "source": "A38",
  "innerRadius": 100,
  "outerRadius": 120,
  "x": 240,
  "y": 320
}
```

3.4.3.2 Arc



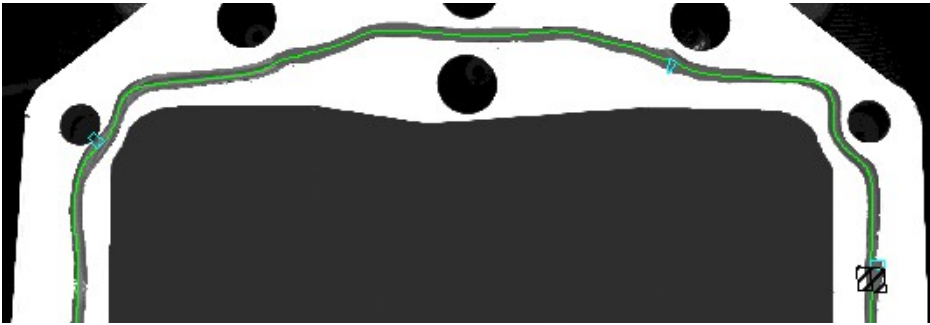
The “PlotArc” function will produce an **Arc** graphic.

Property Name	Type	Default Value	Description
\$type	string	"Arc"	Arc class name.
angle0	float		Angle in degrees to start the arc.
angle1	float		Angle in degrees to end the arc.
radius	float		Radius of the arc.
rotateClockwise	bool		true if the arc is rotated clockwise, false otherwise.
x	float		X co-ordinate of the center of the arc.
y	float		Y co-ordinate of the center of the arc.

Example:

```
{
  "$type": "Arc",
  "angle0": 0,
  "angle1": 90,
  "radius": 300,
  "rotateClockwise": false,
  "x": 400,
  "y": 100
}
```

3.4.3.3 BeadPath



The BeadPath is a specialized graphic associated with the **BeadInspect** and **BeadFind** functions.

Property Name	Type	Default Value	Description
\$type	string	"BeadPath"	BeadPath class name.
caliperWidth	float		Twice the HALF_WIDTH value. Used by GUI to determine the width of the fill edge and inspect areas. - For BeadFind, this is a value from the selected BeadFind result. - For BeadInspect, this is from the spreadsheet function argument.
closed	bool		If true, the path portion of the graphic should be drawn as a closed polyline.
points	float[]		X and Y co-ordinates for all of the vertices in the bead path along with a flag as a flat list of values. Each tuple of values is the x and y co-ordinate for a point, and (possibly) a flag corresponding to that point. If the pointSize is set to 2, this is a 2-tuple that can be indexed at point <i>i</i> as follows: $x = \text{points}[i * 2]$ $y = \text{points}[(i * 2) + 1]$ If the pointSize is set to 3, this is a 3-tuple that can be indexed at point <i>i</i> as follows: $x = \text{points}[i * 3]$ $y = \text{points}[(i * 3) + 1]$ $\text{flag} = \text{points}[(i * 3) + 2]$
pointSize	int	2	Indicates the size of the point structure in the points array. The only supported values are 2 or 3.
showFillAreas	bool		If true, the fill edge areas will be drawn in the output graphic.
showIgnoreAreas	bool		If true, the ignore areas will be drawn in the output graphic.
x	float		X co-ordinate of the location (first point) in the chain.
y	float		Y co-ordinate of the location (first point) in the chain.

3.4.3.4 BlobChain

The chain represents a closed polygon that is used primarily for Blob output graphics. It is similar to the Polygon graphic type, however unlike the Polygon it is immutable and cannot be edited. Typically, Point Chains are much longer than Polygons and can contain hundreds or thousands of points.



An “ExtractBlobs” function will produce a **BlobChain** graphic.

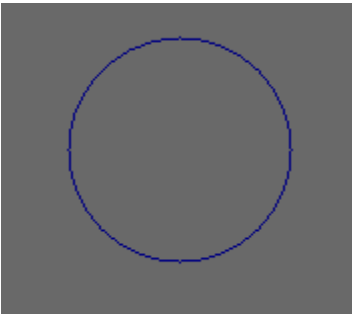
Property Name	Type	Default Value	Description
\$type	string	"BlobChain"	BlobChain class name.
Index	int		Numeric index of the graphic result.
points	float[]		X and Y co-ordinates for all of the vertices in the chain as a flat list of values. Each pair of values is the x and y co-ordinate for a point. In order to find a point at index i: x = points[i*2] y = points[(i *2) + 1]
x	float		X co-ordinate of the location (first point) in the chain.
y	float		Y co-ordinate of the location (first point) in the chain.

Example:

```
{
  "$type": "BlobChain",
  "color": -16744448,
  "source": "A11",
  "index": 0,
  "points": [
    256,
    451,
    ...
  ],
  "x": 256,
  "y": 451
}
```

3.4.3.5 Circle

Basic graphic type used for input/output graphics.



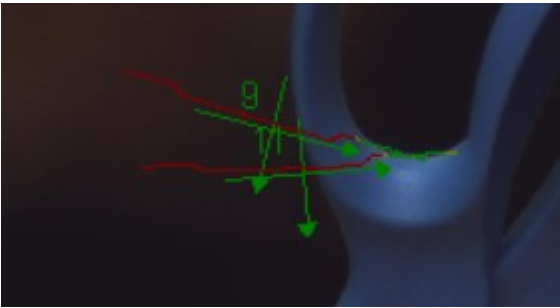
An “EditCircle” or “PlotCircle” function will produce a **Circle** graphic.

Property Name	Type	Default Value	Description
\$type	string	"Circle"	Circle class name.
radius	float		Radius of the circle.
x	float		X co-ordinate of the center of the circle.
y	float		Y co-ordinate of the center of the circle.

Example:

```
{
  "$type": "Circle",
  "color": -16777088,
  "source": "A32",
  "radius": 100,
  "x": 240,
  "y": 320
}
```

3.4.3.6 ColorMatch



This is a graphic that displays line segments of varied colors (e.g. the line segments in the image above). There may be multiple instances of this object that are meant to be rendered together as a single graphic. This graphic is produced by functions such as **FindPatMaxPatterns**.

Property Name	Type	Default Value	Description
\$type	string	"ColorMatch"	ColorMatch class name.
colors	int[]		Array of ARGB colors for each segment. (Red: 0xFFFF0000, Yellow: 0xFFFFF000, Green: 0xFF00FF00)
firstSegmentIndex	int		The index of the first segment of the graphic. (If this graphic does not contain all of the segments, then other graphic objects will contain other portions of this graphic.)
numSegments	int		The number of line segments in this graphic. (This is the number of segments contained in this object.)
segments	float[]		Array of coordinates for each segment in this object. There are 4 times the number of segments values in this array. Each segment is (row0, col0, row1, col1).
totalSegments	int		The total number of line segments in this graphic. If the segments all fit into one object, then this will be the same as numSegments otherwise, it will be a larger number as there will be multiple objects.

Example:

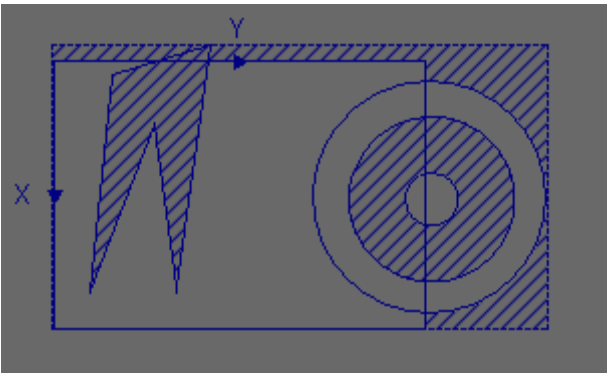
```
{
  "$type": "ColorMatch",
  "color": -16744448,
  "source": "A4",
  "colors": [
    -65536, ...
    -16711936
  ],
  "firstSegmentIndex": 0,
  "numSegments": 67,
  "segments": [
    941.8741455078125, ...
    1365.98291015625
  ],
  "totalSegments": 67
}
```

3.4.3.7 CompositeRegion

Composite regions are a container graphic that cannot be drawn on their own. Instead, they contain a list of graphics that each define how part of the graphic is rendered. The entire composite region is rendered by rendering each sub-region in order.

There are three types of composite regions, which have different rules for the type of sub-region(s) that are allowed. The Composite Region is the most expansive, allowing Annulus, Circle, Polygon, or Region types as sub-regions. The two other composite region types (Masked Region and Polyline Path) only allow Region graphics as sub-regions. All three types have different rules for rendering the sub-regions. The Composite Region classes are used as base classes for the Masked Region and Polyline Path classes in the various implementations of the graphic.

Sub-regions are always contained in a [SubRegion](#) graphic. This is also a container graphic, except for a single graphic that governs how the SubRegion is rendered.



An “EditCompositeRegion” function will produce a **CompositeRegion** graphic.

Property Name	Type	Default Value	Description
\$type	string	"CompositeRegion"	CompositeRegion class name.
subregions	SubRegion[]		Array of SubRegion graphics.

Example:

```
{
  "$type": "CompositeRegion",
  "color": -16777088,
  "source": "A4",
  "subregions": [
    {
      "$type": "SubRegion",
      "color": -16777077,
      "add": true,
      "shape": {
        "$type": "Region",
        "color": -16777088,
        "angle": 0, "curve": 0, "h": 320, "w": 440, "x": 80, "y": 100
      }
    },
    {
      "$type": "SubRegion",
      "color": -16777077,
      "add": false,
      "shape": {
        "$type": "Annulus",
        "color": -16777088,
        "innerRadius": 40,
        "outerRadius": 60.56273651123047,
        "x": 288.94976806640625,
        "y": 409.8630065917969
      }
    }
  ]
}
```

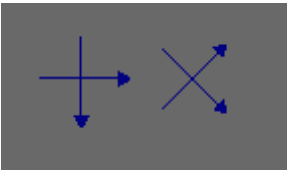
```

{
  "$type": "SubRegion",
  "color": -16777077,
  "add": false,
  "shape": {
    "$type": "Circle",
    "color": -16777088,
    "radius": 53.010, "x": 184.474, "y": 206.027
  }
},
{
  "$type": "SubRegion",
  "color": -16777077,
  "add": false,
  "shape": {
    "$type": "Polygon",
    "color": -16777088,
    "length": 3,
    "points": [303.305, 259.283, 379.305, 217.283, 376.305,
               334.28314208984375
    ]
  }
}
}]
}

```

3.4.3.8 Cross

Basic graphic type used for input/output graphics.



A “PlotCross” function will produce a **Cross** graphic.

Property Name	Type	Default Value	Description
\$type	string	"Cross"	Cross class name.
angle	float	0	Angle of the cross in degrees.
h	float		Height of the cross.
w	float		Width of the cross.
x	float		X co-ordinate for the center of the cross. Note that this is the location where the two arrows actually cross, not the top/left corner.
y	float		Y co-ordinate for the center of the cross. Note that this is the location where the two arrows actually cross, not the top/left corner.

Example:

```
{
  "$type": "Cross",
  "color": -16744448,
  "source": "A5",
  "angle": 0,
  "h": 40,
  "w": 40,
  "x": 100,
  "y": 100
}
```

3.4.3.9 FilledBox

The Filled Box graphic type is a single color solid-fill rectangle, and only drawn under limited circumstances. The camera will send a Filled Box to fill in pixels as an output of Flaw tools when the client requests a reduced image resolution. Flaw tools will use other means (Polyline for example) to display the same graphics when the camera is outputting at full resolution.

Property Name	Type	Default Value	Description
\$type	string	"FilledBox"	FilledBox class name.
h	float		Height of the filled box.
angle	float	0	Rotation angle of the filled box in degrees.
w	float		Width of the filled box.
x	float		X co-ordinate of the location of the filled box.
y	float		Y co-ordinate of the location of the filled box.

Example:

```
{
  "$type": "FilledBox",
  "angle": 0,
  "color": -16777088,
  "source": "A36",
  "h": 100,
  "w": 100,
  "x": 240,
  "y": 320
}
```

3.4.3.10 Fixture

When it is drawn on the screen, the fixture graphic will look very much like the Cross graphic. It differs in that the fixture's purpose is to indicate the location and direction of a co-ordinate system transform in the space of the image. The position of the fixture will change as the user zooms and scrolls the camera image, but the fixture will **not scale with the image** like the cross does. It is a different type of graphic with no width or height properties.



An “Fixture” function will produce a **Fixture** graphic.

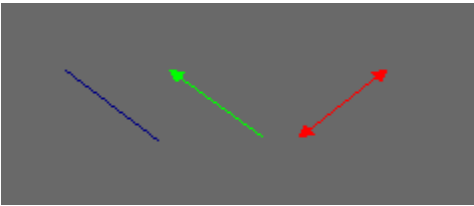
Property Name	Type	Default Value	Description
\$type	string	"Fixture"	Fixture class name.
angle	float		Angle of the fixture in degrees.
x	float		X offset for the fixture.
y	float		Y offset for the fixture.

Example:

```
{
  "$type": "Fixture",
  "color": -16777088,
  "source": "A9",
  "angle": 0,
  "x": 0,
  "y": 0
}
```

3.4.3.11 Line

Basic graphic type used for input/output graphics.



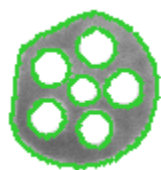
An “EditLine” or “PlotLine” function will produce a **Line** graphic.

Property Name	Type	Default Value	Description
\$type	string	"Line"	Line class name.
adornment0	int	0	Enumeration value for the end adornment of the starting point of the line. 0 means none, 1 means an arrowhead.
adornment1	int	0	Enumeration value for the end adornment of the ending point of the line. 0 means none, 1 means an arrowhead.
x0	float		X co-ordinate for the starting point of the line.
x1	float		X co-ordinate for the ending point of the line.
y0	float		Y co-ordinate for the starting point of the line.
y1	float		Y co-ordinate for the ending point of the line.

Example:

```
{
  "$type": "Line",
  "color": -16777088,
  "source": "A34",
  "x0": 80,
  "x1": 240,
  "y0": 100,
  "y1": 320
}
```

3.4.3.12 LineList

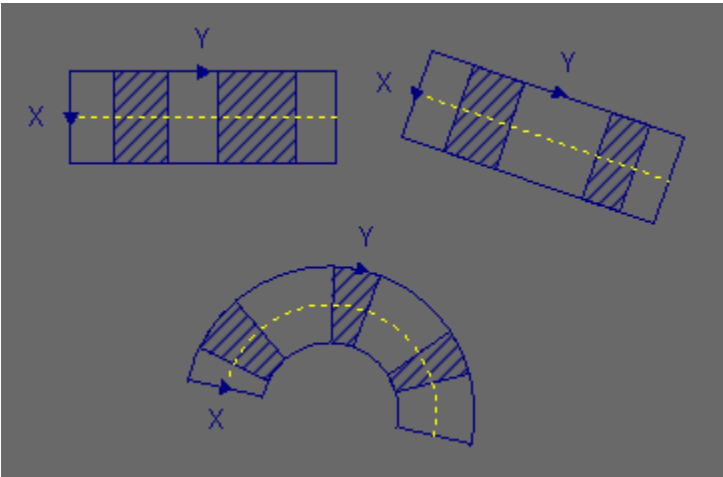


Property Name	Type	Default Value	Description
\$type	string	"LineList"	Polyline class name.
lines	float[]		A list of X0, Y0, X1, Y1 coordinates of one-or-more line segments. This array's length will be a multiple of 4.

Example:

```
{
  "$type": "LineList",
  "color": -16711936,
  "source": "A9",
  "points": [
    112,
    200,
    132,
    200,
    ...]
}
```

3.4.3.13 MaskedRegion



An “EditMaskedRegion” function will produce a **MaskedRegion** graphic.

Property Name	Type	Default Value	Description
\$type	string	"MaskedRegion"	MaskedRegion class name.

Example:

```
{
  "$type": "MaskedRegion",
  "color": -16777088,
  "source": "A2",
  "subregions": [
    {
      "$type": "SubRegion",
      "color": -16777077,
      "add": true,
      "shape": {
        "$type": "Region",
        "color": -16777088,
        "angle": 0,
        "curve": 0,
        "h": 126.86276245117188,
        "w": 440,
        "x": 498.1372375488281,
        "y": 580
      }
    },
    {
      "$type": "SubRegion",
      "color": -16777077,
      "add": false,
      "shape": {
        "$type": "Region",
        "color": -16777088,
        "angle": 0,
        "curve": 0,
        "h": 126.86276245117188,
        "w": 10,
        "x": 498.13720703125,
        "y": 590
      }
    }
  ]
}
```

3.4.3.14 Point

The point graphic is meant to indicate a specific pixel location in the image. The graphics that are drawn for this are statically sized and **do not scale with the image**.



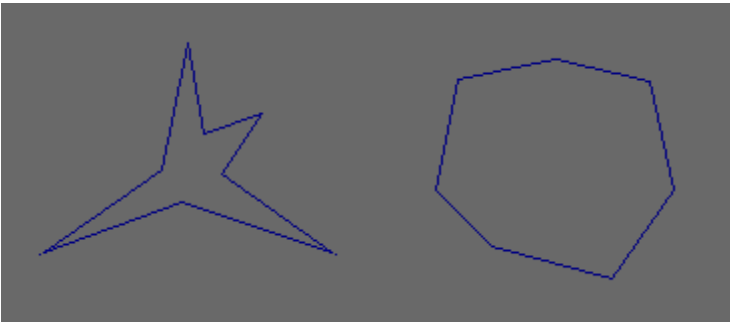
An “EditPoint” or “PlotPoint” function will produce a **Point** graphic.

Property Name	Type	Default Value	Description
\$type	string	"Point"	Point class name.
x	float		X co-ordinate of the point.
y	float		Y co-ordinate of the point.

Example:

```
{
  "$type": "Point",
  "color": -16777088,
  "source": "A36",
  "x": 240,
  "y": 320
}
```

3.4.3.15 Polygon



An “EditPolygon” or “PlotPolygon” function will produce a **Polygon** graphic.

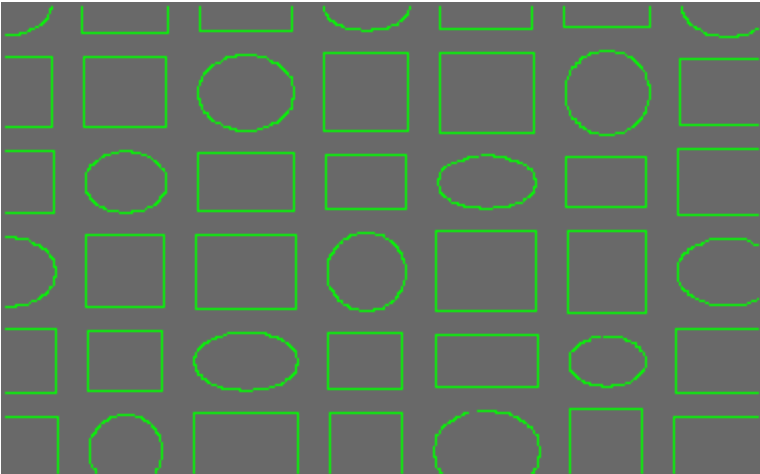
Property Name	Type	Default Value	Description
\$type	string	"Polygon"	Polygon class name.
length	int		Number of vertices in the polygon. Note that this will be half the length of the points array.
points	float[]		X and Y co-ordinates for all of the vertices in the polygon as a flat list of values. Each pair of values is the x and y co-ordinate for a point. In order to find a point at index i: x = points[i*2] y = points[(i*2) + 1]

Example:

```
{
  "$type": "Polygon",
  "color": -16777077,
  "length": 3,
  "points": [
    80,
    320,
    400,
    100,
    400,
    540]
}
```

3.4.3.16 Polyline

This graphic is produced by the Flexible Flaw Tool.

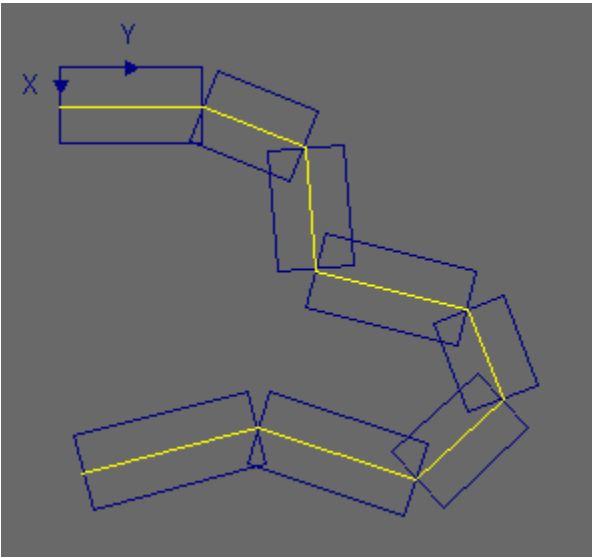


Property Name	Type	Default Value	Description
\$type	string	"Polyline"	Polyline class name.
points	float[]		X and Y co-ordinates for all of the vertices in the polyline as a flat list of values. Each pair of values is the x and y co-ordinate for a point. In order to find a point at index i: x = points[i*2] y = points[(i *2) + 1]

Example:

```
{
  "$type": "Polyline",
  "color": -16711936,
  "source": "A9",
  "points": [
    112,
    200,
    ...]
}
```

3.4.3.17 PolylinePath



An “EditPolylinePath” function will produce a **PolylinePath** graphic.

Property Name	Type	Default Value	Description
\$type	string	"PolylinePath"	PolylinePath class name.

Example:

```
{
  "$type": "PolylinePath",
  "color": -16777088,
  "source": "A5",
  "subregions": [
    {
      "$type": "SubRegion",
      "color": -16777077,
      "add": true,
      "shape": {
        "$type": "Region",
        "color": -16777088,
        "angle": 0,
        "curve": 0,
        "h": 50,
        "w": 220,
        "x": 618.2549438476562,
        "y": 429.068603515625
      }
    },
    {
      "$type": "SubRegion",
      "color": -16777077,
      "add": true,
      "shape": {
        "$type": "Region",
        "color": -16777088,
        "angle": -30,
        "curve": 0,
        "h": 50,
        "w": 220,
        "x": 621.604248046875,
        "y": 661.5684204101562
      }
    }
  ]
}
```

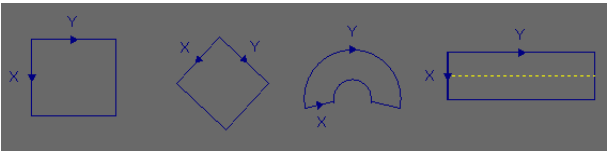
3.4.3.18 Rectangle

Note: In-Sight Cameras do not currently use this shape.

Example:

```
{
  "$type": "Rectangle",
  "color": -16777088,
  "source": "A36",
  "h": 100,
  "w": 100,
  "x": 240,
  "y": 320
}
```

3.4.3.19 Region



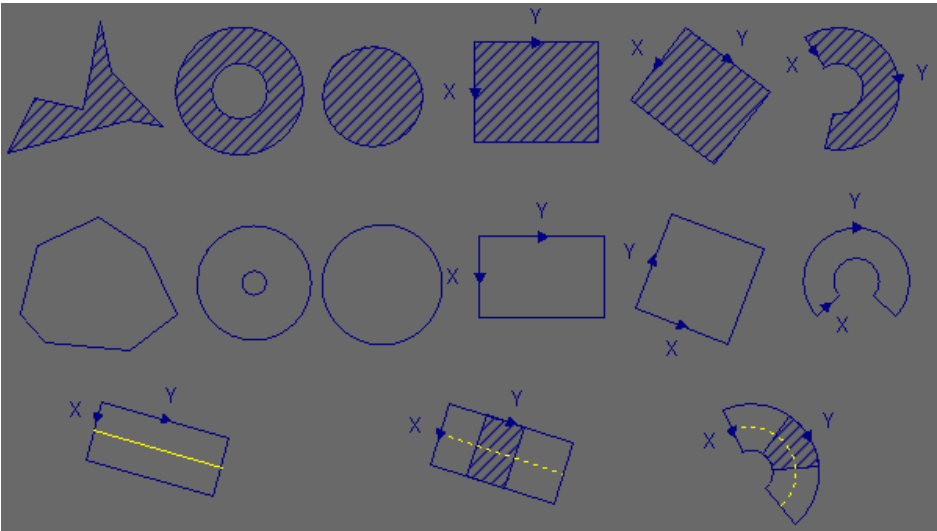
An “EditRegion” or “PlotRegion” function will produce a **Region** graphic.

Property Name	Type	Default Value	Description
\$type	string	"Region"	Region class name.
angle	float	0	Rotation angle of the region in degrees.
curve	float	0	Curve of the region in degrees. On an In-Sight camera, this is the angular deviation between the region's x-axis and the opposing boundary line.
h	float		Height of the region.
showAxesLabels	bool	true	true to draw the X & Y axes labels for the region. false otherwise.
showScanLine	bool	false	true to draw a dotted scan line through the region along its vertical midpoint. false otherwise.
showXArrow	bool	true	true to draw an X axis arrow for the region. false otherwise.
showYArrow	bool	true	true to draw a Y axis arrow for the region. false otherwise.
w	float		Width of the region.
x	float		X co-ordinate of the location of the region.
y	float		Y co-ordinate of the location of the region.

Example:

```
{
  "$type": "Region",
  "color": -16777088,
  "source": "A30",
  "angle": 0,
  "curve": 0,
  "h": 320,
  "w": 440,
  "x": 80,
  "y": 100
}
```

3.4.3.20 SubRegion



The sub-region graphic type is not stand-alone like the other graphics classes. It is valid only in the context of a Composite Region, Masked Region, or Polyline Path. The shapes that are rendered can be a Region, Annulus, Circle, or Polygon that is contained within one of the previously given graphics. All of the possible permutations for these shapes can also be used in a sub-region, as well as versions where the shapes are shaded (for subtracted sub-regions in the composite region). Note that a sub-region graphic itself will never be drawn. Clients will draw the graphic contained in the sub-region shape property.

Property Name	Type	Default Value	Description
\$type	string	"SubRegion"	SubRegion class name.
add	bool		true if the subregion is added to the composite region. false if it is subtracted from the composite region. NOTE: When a SubRegion is subtracted, it is typically rendered either as cross-hatch or shaded to designate that it is ignored.
shape	ShapeBase		Shape object for the sub-region. This will be one of: Annulus, Circle, Polygon, or Region.

3.4.3.21 Text



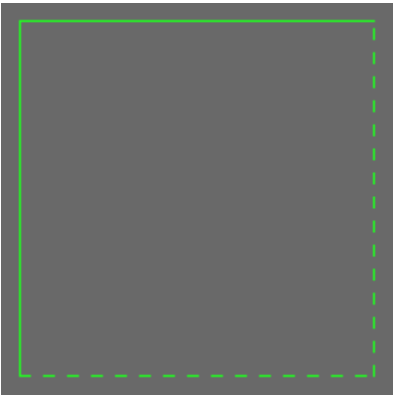
A “PlotString” function will produce a **Text** graphic.

Property Name	Type	Default Value	Description
\$type	string	"Text"	Text class name.
bgColor	int	0x00FFFFFF	32-bit ARGB value of the background color for the text. The default value is transparent (0x00FFFFFF). This field is not defined when it is the default value. This color should be used for the background rectangle behind the text (when not transparent). NOTE: The foreground color of the text is return as the “color” field (as is the case for the other graphics).
text	string		Text string to display.
x	float		X co-ordinate of the top/left corner of the text.
y	float		Y co-ordinate of the top/left corner of the text.

Example:

```
{
  "$type": "Text",
  "color": -16744448,
  "fontSize": 12,
  "source": "B1",
  "text": "104",
  "x": 10,
  "y": 300
}
```

3.4.3.22 4Side



This is an output graphic to show the location of a QR code. It is rendered with a dashed line between the 2nd and 3rd and the 3rd and 4th points.

The “ReadIDMax” and “ReadIDCode” functions will produce **4Side** graphics.

Property Name	Type	Default Value	Description
\$type	string	"4Side"	4Side class name.
x0	float		X co-ordinate of the 1 st point in the 4-sided polygon.
x1	float		X co-ordinate of the 2 nd point in the 4-sided polygon.
x2	float		X co-ordinate of the 3 rd point in the 4-sided polygon.
x3	float		X co-ordinate of the 4 th point in the 4-sided polygon.
y0	float		Y co-ordinate of the 1 st point in the 4-sided polygon.
y1	float		Y co-ordinate of the 2 nd point in the 4-sided polygon.
y2	float		Y co-ordinate of the 3 rd point in the 4-sided polygon.
y3	float		Y co-ordinate of the 4 th point in the 4-sided polygon.

Example:

```
{
  "$type": "4Side",
  "color": -16777088,
  "source": "A36",
  "x0": 100,
  "x1": 200,
  "x2": 200,
  "x3": 100,
  "y0": 100,
  "y1": 100,
  "y2": 200,
  "y3": 200
}
```

3.5 Cell Results

Each cell result that may be part of the "cells" array in the **HmiResult** will have these common properties:

Property	Description
\$type	The specific type of the cell.
data	The value of the cell. This varies by cell type and may be a simple value or complex object.
disabled	Designates that the cell is not enabled. This typically occurs when a cell has been explicitly disabled via the Cell State properties in ISE. (This is false, by default, and thus not part of the object unless true.) A web HMI should draw an item that is disabled in "grayed-out" text and should not allow it to be edited.
editable	Designates that the cell is an editable type (e.g. EditInt, EditRegion, etc.). This was added for version 2.0 of the protocol. (This is false, by default, and thus not part of the object unless true.)
error	Designates that the cell is in the error state. This occurs when a tool or expression in a job cannot be evaluated or fails. (This is false, by default, and thus not part of the object unless true.) A web HMI should draw an item that is in the error state as "Error" or "#ERR" and not allow it to be edited.
location	The cell location will be defined for every cell.
name	The name of the cell. Depending upon how the session results were requested, not every cell may have a name. (NOTE: All cells in an EasyView will have cell names.)

The following cell results may be part of the cell array in the result. *Note: The following cell results show examples of enabled cells, so the "disabled" field will not be present.*

Note: Beginning with the API revision 3.0, some new devices will return the following cell types prefixed with "Hmi".

3.5.1 FloatResult / HmiFloatResult

This is a numeric result object that is returned for cells that contain numbers.

Property	Description
\$type	"FloatResult" or "HmiFloatResult"
data	The data will be a number.

Example:

```
{
  "$type": "FloatResult",
  "data": 123.0,
  "location": "A3",
  "name": "NamedCell1"
}
```

3.5.2 StringResult / HmiStringResult

This is a string result object that is returned for cells that contain strings (e.g. 'hello').

Property	Description
\$type	"StringResult" or "HmiStringResult"
data	The data will be a string.

Example:

```
{
  "$type": "StringResult",
  "data": "MyString",
  "location": "A3",
  "name": "NamedCell1"
}
```

3.5.3 EditFloatResult / HmiEditFloatResult

This is a result for a cell that contains an "EditFloat" function.

Property	Description
\$type	"EditFloatResult" or "HmiEditFloatResult"
data	The data will be a number.
min	The minimum number that is valid for the cell.
max	The maximum number that is valid for the cell.

Example:

```
{
  "$type": "EditFloatResult",
  "data": 123.0,
  "location": "A3",
  "name": "NamedCell1",
  "max": 200.0, "min": 0.0
}
```

3.5.4 EditIntResult / HmiEditIntResult

This is a result for a cell that contains an "EditInt" function.

Property	Description
\$type	"EditIntResult" or "HmiEditIntResult"
data	The data will be a number.
min	The minimum number that is valid for the cell.
max	The maximum number that is valid for the cell.

Example:

```
{
  "$type": "EditIntResult",
  "data": 123,
  "location": "A3",
  "name": "NamedCell1",
  "max": 200, "min": 0
}
```

3.5.5 EditStringResult / HmiEditStringResult

This is a result for a cell that contains an "EditString" function.

Property	Description
\$type	"EditStringResult" or "HmiEditStringResult"
data	The data will be a string.
maxLength	The maximum length of the string.
maskInput	A boolean flag that designates whether the value should be masked as it is entered. (Masked text entry is typically displayed as asterisks.)

Example:

```
{
  "$type": "EditStringResult",
  "data": "MyString",
  "location": "A3",
  "name": "NamedCell1",
  "maskInput": true, // This is false by default
  "maxLength": 20
}
```

3.5.6 HmiButtonResult

This is a result for a cell that contains a "Button" function or any Edit Graphics cell that would be rendered as a Button (i.e. **EditRegion**, **EditCircle**, etc.). This is received from devices that use the protocol versions 3.0 or after.

Property	Description
\$type	"HmiButtonResult"
data	This value will be a number (0) for a regular spreadsheet Button, or it will be the value of the Graphic, if it refers to an editable graphic. Possible graphic types include: "Region", "Circle", "Line", "Point", "Annulus", "Polygon", and "CompositeRegion". Note that the "ComposteRegion" type is returned for EditCompositRegion, EditPolyLine, and EditMaskedRegion functions.
caption	The text to be shown on the button.

Examples:

Regular Button Result:

```
{
  "$type": "HmiButtonResult",
  "data": 0,
  "name": "NamedCell1",
  "location": "A3",
  "caption": "Button Text"
}
```

EditRegion Result:

```
{
  "$type": "HmiButtonResult",
  "data": {
    "$type": "Region",
    "angle": 0, "curve": 0, "h": 320, "w": 440, "x": 580, "y": 440
  },
  "name": "NamedCell1",
  "location": "A3",
  "caption": "Edit Region Button"
}
```

3.5.7 CheckBoxResult / HmiCheckBoxResult

This is a result for a cell that contains a "CheckBox" function.

Property	Description
\$type	"CheckBoxResult" or "HmiCheckBoxResult"
data	The value will be 0 when unchecked and 1 when checked.
caption	The text to be shown for the checkbox control.

Example:

```
{
  "$type": "CheckBoxResult",
  "data": 0,
  "name": "NamedCell1",
  "location": "A3",
  "caption": "CheckBox Text"
}
```

3.5.8 ListBoxResult / HmiListBoxResult

This is a result for a cell that contains a "ListBox" function.

Property	Description
\$type	"ListBoxResult" or "HmiListBoxResult"
data	The numeric index into the options that is the value for this cell.
options	The array of strings that are used to display the current selected index or multiple values in a drop-down control.

Example:

```
{
  "$type": "ListBoxResult",
  "data": 0, // This is the index into the "options"
  "name": "NamedCell1",
  "location": "A3",
  "options": ["opt1", "opt2", "opt3"]
}
```

3.5.9 HmiStatusResult

This is a result for a cell that contains a "Status" function for protocol revision 3.0 and after.

Property	Description
\$type	"HmiStatusResult"
data	The numeric value of the status. An HMI will typically not need to use this value.
color	This control will typically be rendered as a circle of this color in an HMI.
caption	The caption that should be rendered next to the status circle.

Example:

```
{
  "$type": "StatusLightResult",
  "data": 0,
  "name": "NamedCell1",
  "location": "A3",
  "caption": "ZeroValueCaption",
  "color": 4278255360 // ARGB
}
```

3.5.10 MultiStatusResult / HmiMultiStatusResult

This is a result for a cell that contains a "MultiStatus" function.

Property	Description
\$type	"MultiStatusResult" or "HmiMultiStatusResult"
data	The numeric value of the status.
color0	The color value that should be used to render bits of value 0.
color1	The color value that should be used to render bits of value 1.
numBits	The number of bits used in the status. In the HMI, there will typically be a status circle drawn for each bit.
reverse	The direction of the bit values. This is false by default, designating that the first bit is lowest ordered bit.
startBit	The first bit to be used in the data value. (i.e. shift the value 1 in the value order direction to get to the first bit value)

Example:

```
{
  "$type": "MultiStatusResult",
  "color0": 4294967040, // This is an ARGB color bit value 0
  "color1": 4278255360, // This is an ARGB color bit value 1
  "data": 0,
  "location": "A22",
  "name": "NamedCell1",
  "numBits": 10, // The number of value bits to examine
  "reverse": 0,
  "startBit": 0 // The start bit for the value
}
```

3.5.11 ColorLabelResult / HmiColorLabelResult

This is a result for a cell the contains a "ColorLabel" function.

Property	Description
\$type	"ColorLabelResult" or "HmiColorLabelResult"
data	The string to display as the value.
backColor	The background color that may be used for rendering.
foreColor	The foreground color that may be used for rendering.

Example:

```
{
  "$type": "ColorLabelResult",
  "data": "MyColorLabel",
  "backColor": 3452816845, // ARGB
  "foreColor": 4278255360, // ARGB
  "location": "A26",
  "name": "NamedCell11"
}
```

3.5.12 HmiProfileViewResult

This is a result for a cell that contains a "GetProfile" function. This result type was introduced in protocol version 3.0.

Property	Description
\$type	"HmiProfileViewResult"
data	The string to display as the value.
viewName	The string that identifies the view related to this result.

Example:

```
{
  "$type": "HmiProfileViewResult",
  "location": "G4",
  "name": "Cell_G4",
  "viewName": "Cell_G4View",
  "data": "□Profile3D"
}
```

3.5.13 ErrorCellResult / HmiErrorCellResult

This cell result is returned when a cell has an error result. A web HMI should render a cell result of this type as an error.

Example:

```
{
  "$type": "ErrorCellResult",
  "location": "A25",
  "name": "NamedCell11"
}
```

3.5.14 UnsupportedCellResult / HmiUnsupportedCellResult

This cell result is returned for cell types that are not currently supported in the web HMI. (For example, the “Chart” function cell.) A web HMI should either not render a cell result of this type or render it as not supported.

Example:

```
{
  "$type": "UnsupportedCellResult",
  "location": "A25",
  "name": "NamedCell1"
}
```

3.5.15 DialogResult

This is a dialog result object that is returned for cells that contain a Dialog function. Typically, a cell like this will be rendered as a button.

Property	Description
\$type	"DialogResult"
data	The data will be a string.

Example:

```
{
  "$type": "DialogResult",
  "data": "MyDialog",
  "location": "A3",
  "name": "NamedCell1"
}
```

3.5.16 Deprecated Cell Result Types

These types were returned from devices that use the protocol versions prior to 3.0. Future cameras are not expected to use these types.

3.5.16.1 ButtonResult

This is a result for a cell that contains a "Button" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, an **HmiButtonResult** will be included in the results for this cell.

Property	Description
\$type	"ButtonResult"
data	Always zero. (unused)
caption	The text to be shown on the button.

Example:

```
{
  "$type": "ButtonResult",
  "data": 0, // This is always zero
  "name": "NamedCell1",
  "location": "A3",
  "caption": "Button Text"
}
```

3.5.16.2 EditAnnulusResult

This is a result for a cell that contains an "EditAnnulus" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, an **HmiButtonResult** will be included in the results for this cell.

Property	Description
\$type	"EditAnnulusResult"
data	The data will be an Annulus object.
caption	A label that can be displayed to identify the graphic.
move	A flag that designates whether the graphic should be moved (i.e. draggable) in the HMI.
resize	A flag that designates whether the graphic should be resized in the HMI.

Example:

```
{
  "$type": "EditAnnulusResult",
  "location": "A30",
  "data": {
    "$type": "Annulus", "innerRadius": 100, "outerRadius": 200, "x": 80, "y": 100
  },
  "caption": "MyCaption",
  "move": true,
  "name": "NamedCell1",
  "resize": true
}
```

3.5.16.3 EditPolygonResult

This is a result for a cell that contains an "EditPolygon" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, an **HmiButtonResult** will be included in the results for this cell.

Property	Description
\$type	"EditPolygonResult"
data	The data will be a Polygon object.
caption	A label that can be displayed to identify the graphic.
addPoints	A flag that designates whether points should be addable to the graphic in the HMI.
move	A flag that designates whether the graphic should be moved (i.e. draggable) in the HMI.
movePoints	A flag that designates whether the graphic's point should be moved individually (i.e. draggable) in the HMI.
removePoints	A flag that designates whether points should be removable to the graphic in the HMI.
rotate	A flag that designates whether the graphic should be rotated in the HMI.
scale	A flag that designates whether the graphic should be scale adjustable in the HMI.

Example:

```
{
  "$type": "EditPolygonResult",
  "location": "A30",
  "data": {
    "$type": "Polygon",
    "length": 3,
    "points": [10, 10, 10, 50, 20, 50]
  },
  "caption": "MyCaption",
  "addPoints": true, "move": true, "movePoints": true,
  "name": "NamedCell1",
  "removePoints": true, "rotate": true, "scale": true
}
```

3.5.16.4 EditCompositeRegionResult

This is a result for a cell that contains an "EditCompositeRegion" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, an **HmiButtonResult** will be included in the results for this cell.

Property	Description
\$type	"EditCompositeRegionResult"
data	The data will be a CompositeRegion object.
caption	A label that can be displayed to identify the graphic.
addRegion	A flag that designates whether a SubRegion should be addable to the graphic in the HMI.
editRegion	A flag that designates whether a SubRegion should be editable in the graphic in the HMI.
move	A flag that designates whether the graphic should be moved (i.e. draggable) in the HMI.
removeRegion	A flag that designates whether a SubRegion should be removable from the graphic in the HMI.
rotate	A flag that designates whether the graphic should be rotated in the HMI.

Example:

```
{
  "$type": "EditCompositeRegionResult",
  "location": "A30",
  "data": {
    "$type": "CompositeRegion",
    ...
  },
  "caption": "MyCaption",
  "addRegion": true,
  "editRegion": true,
  "move": true,
  "name": "NamedCell1",
  "removeRegion": true,
  "rotate": true
}
```

3.5.16.5 EditMaskedRegionResult

This is a result for a cell that contains an "EditMaskedRegion" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, an **HmiButtonResult** will be included in the results for this cell.

The **MaskedRegion** object that is held in the data member holds an array of one or more **SubRegion** objects.

Property	Description
\$type	"EditMaskedRegionResult"
data	The data will be a MaskedRegion object.
caption	A label that can be displayed to identify the graphic.
addRegion	A flag that designates whether a SubRegion should be addable to the graphic in the HMI.
bend	A flag that designates whether the graphic should be bendable in the HMI.
editRegion	A flag that designates whether a SubRegion should be editable in the graphic in the HMI.
move	A flag that designates whether the graphic should be moved (i.e. draggable) in the HMI.
removeRegion	A flag that designates whether a SubRegion should be removable from the graphic in the HMI.
resize	A flag that designates whether the graphic should be resized in the HMI.
rotate	A flag that designates whether the graphic should be rotated in the HMI.

Example:

```
{
  "$type": "EditMaskedRegionResult",
  "location": "A30",
  "data": {
    "$type": "MaskedRegion",
    "subregions": [
      {"$type": "SubRegion", "add": true, "shape":
      {"$type": "Region", "angle": 0.0, "curve": 0.0, "h": 50.0, "w": 440.0, "x": 215.0, "y": 100.0}},
      {"$type": "SubRegion", "add": false, "shape":
      {"$type": "Region", "angle": 0.0, "curve": 0.0, "h": 50.0, "w": 10.0, "x": 215.0, "y": 110.0}}
    ]
  },
  "caption": "MyCaption",
  "addRegion": true,
  "bend": true,
  "editRegion": true,
  "move": true,
  "name": "NamedCell1",
  "removeRegion": true,
  "resize": true,
  "rotate": true
}
```

3.5.16.6 EditPolylinePathResult

This is a result for a cell that contains an "EditPolylinePath" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, an **HmiButtonResult** will be included in the results for this cell.

Property	Description
\$type	"EditPolylinePathResult"
data	The data will be a PolylinePath object.
caption	A label that can be displayed to identify the graphic.
addPoints	A flag that designates whether points should be addable to the graphic in the HMI.
move	A flag that designates whether the graphic should be moved (i.e. draggable) in the HMI.
movePoints	A flag that designates whether the graphic's point should be moved individually (i.e. draggable) in the HMI.
removePoints	A flag that designates whether points should be removable to the graphic in the HMI.
rotate	A flag that designates whether the graphic should be rotated in the HMI.
scale	A flag that designates whether the graphic should be scale adjustable in the HMI.

Example:

```
{
  "$type": "EditPolylinePathResult",
  "location": "A30",
  "data": {
    "$type": "PolylinePath",
    ...
  },
  "caption": "MyCaption",
  "addPoints": true,
  "move": true,
  "movePoints": true,
  "name": "NamedCell1",
  "removePoints": true,
  "rotate": true,
  "scale": true
}
```

3.5.16.7 EditRegionResult

This is a result for a cell that contains an "EditRegion" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, an **HmiButtonResult** will be included in the results for this cell.

Property	Description
\$type	"EditRegionResult"
data	The data will be a Region object.
caption	A label that can be displayed to identify the graphic.
bend	A flag that designates whether the graphic should be curve adjusted in the HMI.
move	A flag that designates whether the graphic should be moved (i.e. draggable) in the HMI.
resize	A flag that designates whether the graphic should be resized in the HMI.
rotate	A flag that designates whether the graphic should be rotated in the HMI.

Example:

```
{
  "$type": "EditRegionResult",
  "location": "A30",
  "bend": true,
  "caption": "MyCaption",
  "data": {
    "$type": "Region",
    "angle": 0, "curve": 0, "h": 320, "w": 440, "x": 80, "y": 100
  },
  "move": true, "name": "NamedCell1", "resize": true, "rotate": true
}
```

3.5.16.8 EditCircleResult

This is a result for a cell that contains an "EditCircle" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, **HmiButtonResult** will be included in the results for this cell.

Property	Description
\$type	"EditCircleResult"
data	The data will be a Circle object.
caption	A label that can be displayed to identify the graphic.
move	A flag that designates whether the graphic should be moved (i.e. draggable) in the HMI.
resize	A flag that designates whether the graphic should be resized in the HMI.

Example:

```
{
  "$type": "EditCircleResult",
  "location": "A30",
  "data": {
    "$type": "Circle",
    "radius": 100, "x": 80, "y": 100
  },
  "caption": "MyCaption", "move": true, "name": "NamedCell1", "resize": true
}
```

3.5.16.9 EditLineResult

This is a result for a cell that contains an "EditLine" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, an **HmiButtonResult** will be included in the results for this cell.

Property	Description
\$type	"EditLineResult"
data	The data will be a Line object.
caption	A label that can be displayed to identify the graphic.
move	A flag that designates whether the graphic should be moved (i.e. draggable) in the HMI.
move0	A flag that designates whether the first point can be moved individually in the HMI.
move1	A flag that designates whether the second point can be moved individually in the HMI.
resize	A flag that designates whether the graphic should be resized in the HMI.

Example:

```
{
  "$type": "EditLineResult",
  "location": "A30",
  "data": {
    "$type": "Line",
    "x0": 10, "y0": 10, "x1": 100, "y1": 100
  },
  "caption": "MyCaption", "move": true, "move0": true, "move1": true,
  "name": "NamedCell1", "resize": true
}
```

3.5.16.10 EditPointResult

This is a result for a cell that contains an "EditPoint" function. This is received from devices that use the protocol versions prior to 3.0, otherwise, an **HmiButtonResult** will be included in the results for this cell.

Property	Description
\$type	"EditPointResult"
data	The data will be a Point object.
caption	A label that can be displayed to identify the graphic.
move	A flag that designates whether not the graphic should be moved (i.e. draggable) in the HMI.

Example:

```
{
  "$type": "EditPointResult",
  "location": "A30",
  "data": {
    "$type": "Point",
    "x": 10, "y": 10, "x1": 100, "y1": 100
  },
  "caption": "MyCaption", "move": true, "name": "NamedCell1"
}
```

3.5.16.11 StatusResult

This is a result for a cell that contains a "Status" function prior to protocol revision 3.0.

Property	Description
\$type	"StatusResult"
data	The numeric value of the status. The control should be rendered as a red (negative value), yellow (zero), or a green (positive value) status circle.
caption	The caption that may be shown for the status.

Example:

```
{
  "$type": "StatusResult",
  "data": 0,
  "name": "NamedCell1",
  "location": "A3",
  "caption": "MyCaption"
```

3.5.16.12 StatusLightResult

This is a result for a cell that contains a "StatusLight" function.

Property	Description
\$type	"StatusLightResult"
data	The numeric value of the status. An HMI will typically not need to use this value.
color	This control will typically be rendered as a circle of this color in an HMI.
caption	The caption that should be rendered next to the status circle.

Example:

```
{
  "$type": "StatusLightResult",
  "data": 0,
  "name": "NamedCell1",
  "location": "A3",
  "caption": "ZeroValueCaption",
  "color": 4278255360 // ARGB
}
```

3.6 HmiRqState

This object holds the state of the result queue.

Property	Description
\$type	"HmiRqState"
frozen	This is a boolean value that represents the frozen state of the result queue. The default value is false.
ids	An array of integers that designate the id of each queued result. Note: The ids are assigned sequentially as results are added to the queue. The default value is empty and will not be returned with this object when the queue is not frozen or empty.
slotIndex	The zero-based index of the slot for that the HmiResult holds. This will be -1 if the queue is not frozen or there are no results in it.
slots	The number of filled slots (i.e. results) in the queue.
status	An array of integers that designate pass/fail status for each queued result. The possible values are: 0 = undefined, 1 = pass, 2 = fail The default value is empty and will not be returned with this object when the queue is not frozen or empty.

Example:

```
{
  "$type": "HmiRqState",
  "frozen": false,
  "slotIndex": -1,
  "slots": 0
}
```

3.7 HmiRq

This object allows the camera's result queue to be examined and manipulated.

Property	Description
\$type	"HmiRq"
frozen	Gets or sets the frozen state of the queue. When the queue is first frozen, the slotIndex will be set to the last item in the queue.
rejectAction	Gets or sets the action that should occur when the camera encounters a reject (i.e. failed result). This defaults to "none", but it may be set to "freeze".
slotIndex	The zero-based slot index of the slot selected for the HmiSession . Valid values are from 0 to the number of queued results minus 1.

Method	Description
deleteAllSlots	Clears the entire queue, removing all of the results.
deleteSlot	Removes an individual item from the queue. The zero-based index of the item into the frozen queue should be used to designate the result to remove.
getAllImageUrls	Gets an array that includes URLs for all the images held in results in the frozen queue.
getImageUrl	Gets a URL for an individual results image from the queue. The zero-based index of the item into the frozen queue should be used to designate the result.
saveImageFtp	This method saves the image/s from the frozen queue to an FTP server. The required arguments are as follows: • url: The URL (including the filename without extension) used to access the FTP server. • ftpUserName: The base64 ASCII-encoded username used to access the FTP server. • ftpUserPassword: The base64 ASCII-encoded password used to access the FTP server. • slotIndex: The zero-based slot index in the queue to save or -1 to save the entire queue. • encodedArgs: The username and password are typically base64 encoded. Use false to pass in unencoded username and password strings. Note that a user must be logged in to execute this method. Also, an editor (i.e. ISE or VisionView) must not be connected.

4. Error Handling

For PUT and POST requests where there is no expected return value, on success, a response without a “body” is returned. For example:

```
{
  "$type": "resp",
  "id": 1
}
```

When there is an error completing the request, an error response is returned:

```
{
  "$type": "resp",
  "id": 1,
  "error": -1610311888,
  "body": "Access denied"
}
```

If an end point does not exist, an error response is returned:

```
{
  "$type": "resp",
  "id": 1,
  "error": -2147477839,
  "body": "Member not found"
}
```

Sometimes an **HmiError** object will be used to convey additional information:

```
{
  "$type": "HmiError",
  "errorCode": 1,
  "message": "Access denied",
  "status": -1610311888
}
```

4.1 HTTP Responses

If you are making HTTP requests for the resources, then the expected response would look like this...

A successful GET:

http://127.0.0.1:80/cam0/hmi/info

200 OK Response with JSON payload in the body

When a page not found:

http://127.0.0.1:80/cam0/hmi/info2

HTTP ERROR 404: page can't be found

Access to a root resource besides 'cam0/hmi':

http://127.0.0.1:80/hmi

HTTP ERROR 403: Access was denied

Appendix A: Editable Graphic Buttons

Here is an example of the JSON for the edit graphic type **ButtonResults** for the protocol revision 3.0 and after:

<pre>Region: { "\$type": "HmiButtonResult", "editable": true, "location": "A14", "name": "EditRegion", "caption": "MyEditRegion", "data": { "\$type": "Region", "color": -16777077, "font": "Arial", "fontSize": 9, "graphicId": -1, "lineThickness": 2, "runtimeEditable": false, "source": "", "angle": 0.0, "curve": 0.0, "h": 320.0, "showAxesLabels": true, "showScanLine": false, "showXArrow": true, "showYArrow": true, "w": 440.0, "x": 500.0, "y": 380.0 } }</pre>	<pre>Circle: { "\$type": "HmiButtonResult", "editable": true, "location": "A16", "name": "EditCircle", "caption": "MyEditCircle", "data": { "\$type": "Circle", "color": -16777077, "font": "Arial", "fontSize": 9, "graphicId": -1, "lineThickness": 2, "runtimeEditable": false, "source": "", "radius": 100.0, "x": 720.0, "y": 540.0 } }</pre>
<pre>Line: { "\$type": "HmiButtonResult", "editable": true, "location": "A17", "name": "EditLine", "caption": "MyEditLine", "data": { "\$type": "Line", "color": -16777077, "font": "Arial", "fontSize": 9, "graphicId": -1, "lineThickness": 2, "runtimeEditable": false, "source": "", "adornment0": 0, "adornment1": 0, "x0": 500.0, "x1": 720.0, "y0": 380.0, "y1": 540.0 } }</pre>	<pre>Point: { "\$type": "HmiButtonResult", "editable": true, "location": "A21", "name": "EditPoint", "caption": "MyEditPoint", "data": { "\$type": "Point", "color": -16777077, "font": "Arial", "fontSize": 9, "graphicId": -1, "lineThickness": 2, "runtimeEditable": false, "source": "", "x": 720.0, "y": 540.0 } }</pre>

Annulus: <pre> { "\$type": "HmiButtonResult", "editable": true, "location": "A18", "name": "EditAnnulus", "caption": "MyEditAnn", "data": { "\$type": "Annulus", "color": -16777077, "font": "Arial", "fontSize": 9, "graphicId": -1, "lineThickness": 2, "runtimeEditable": false, "source": "", "innerRadius": 100.0, "outerRadius": 120.0, "x": 720.0, "y": 540.0 } } </pre>	Polygon: <pre> { "\$type": "HmiButtonResult", "editable": true, "location": "A19", "name": "EditPolygon", "caption": "MyEditPolygon", "data": { "\$type": "Polygon", "color": -16777077, "font": "Arial", "fontSize": 9, "graphicId": -1, "lineThickness": 2, "runtimeEditable": false, "source": "", "length": 3, "points": [720.0, 380.0, 500.0, 700.0, 940.0, 700.0] } } </pre>
---	--

CompositeRegion (for EditCompositeRegion):

```
{
  "$type": "HmiButtonResult",
  "editable": true,
  "location": "A15",
  "name": "EditComposite",
  "caption": "MyEditComposite",
  "data": {
    "$type": "CompositeRegion",
    "color": -16777077,
    "font": "Arial",
    "fontSize": 9,
    "graphicId": -1,
    "lineThickness": 2,
    "runtimeEditable": false,
    "source": "",
    "subregions": [{
      "$type": "SubRegion",
      "color": -16777077,
      "font": "Arial",
      "fontSize": 9,
      "graphicId": -1,
      "lineThickness": 2,
      "runtimeEditable": false,
      "source": "",
      "add": true,
      "shape": {
        "$type": "Region",
        "color": -16777077,
        "font": "Arial",
        "fontSize": 9,
        "graphicId": -1,
        "lineThickness": 2,
        "runtimeEditable": false,
        "source": "",
        "angle": 0.0,
        "curve": 0.0,
        "h": 320.0,
        "showAxesLabels": true,
        "showScanLine": false,
        "showXArrow": true,
        "showYArrow": true,
        "w": 440.0,
        "x": 500.0,
        "y": 380.0
      }
    }]
  }
}
```

CompositeRegion (for EditPolyline):

```
{
  "$type": "HmiButtonResult",
  "editable": true,
  "location": "A20",
  "name": "EditPolyline",
  "caption": "MyEditPolyline",
  "data": {
    "$type": "CompositeRegion",
    "color": -16777077,
    "font": "Arial",
    "fontSize": 9,
    "graphicId": -1,
    "lineThickness": 2,
    "runtimeEditable": false,
    "source": "",
    "subregions": [{
      "$type": "SubRegion",
      "color": -16777077,
      "font": "Arial",
      "fontSize": 9,
      "graphicId": -1,
      "lineThickness": 2,
      "runtimeEditable": false,
      "source": "",
      "add": true,
      "shape": {
        "$type": "Region",
        "color": -16777077,
        "font": "Arial",
        "fontSize": 9,
        "graphicId": -1,
        "lineThickness": 2,
        "runtimeEditable": false,
        "source": "",
        "angle": 0.0,
        "curve": 0.0,
        "h": 50.0,
        "showAxesLabels": true,
        "showScanLine": false,
        "showXArrow": true,
        "showYArrow": true,
        "w": 220.0,
        "x": 508.0,
        "y": 461.5
      }
    }]
  }
}
```

CompositRegion (for EditMaskedRegion):

```
{
  "$type": "HmiButtonResult",
  "editable": true,
  "location": "A22",
  "name": "EditMaskedRegion",
  "caption": "EditMaskedRegion",
  "data": {
    "$type": "CompositeRegion",
    "color": -16777077,
    "font": "Arial",
    "fontSize": 9,
    "graphicId": -1,
    "lineThickness": 2,
    "runtimeEditable": false,
    "source": "",
    "subregions": [{
      "$type": "SubRegion",
      "color": -16777077,
      "font": "Arial",
      "fontSize": 9,
      "graphicId": -1,
      "lineThickness": 2,
      "runtimeEditable": false,
      "source": "",
      "add": true,
      "shape": {
        "$type": "Region",
        "color": -16777077,
        "font": "Arial",
        "fontSize": 9,
        "graphicId": -1,
        "lineThickness": 2,
        "runtimeEditable": false,
        "source": "",
        "angle": 0.0,
        "curve": 0.0,
        "h": 50.0,
        "showAxesLabels": true,
        "showScanLine": false,
        "showXArrow": true,
        "showYArrow": true,
        "w": 440.0,
        "x": 500.0,
        "y": 515.0
      }
    }]
  }
}
```