



Impact Software HTTP Access

Revision H

Impact Software HTTP Access

This document describes the HTTP commands used to get or set data on a Datalogic vision device over a local area network. These commands can be sent to the vision device using a standard Internet browser or programmatically, using Visual Basic, for example.

NOTES: The Vision Program Manager (VPM) user interface was enhanced in version 9.0. Because of these changes, some of the system object names and icons in the VPM task tree have changed. The command format remains the same. These discrepancies are noted where applicable.

All commands and names (including Vision System, vision system object, program, task, and tool names) are case sensitive.

Getting Data

The general command format to get data from the device has two forms, depending whether you are getting data from a vision system object or a task tool.

To get an image from the vision device, see Getting an image on page 8.

Getting Vision System Object Data

The command format to get data from a vision system object is:

`http://ip address/getData/Vision System.vision system object name:port name`

where

ip address is the IP Address of the vision device

getData is the command to get data from the device

Vision System is the two words Vision System (even though this is displayed as Impact in 9.0 and later)

vision system object name is the name of the vision system object

port name is the name of the property (port) (Note that a colon separates the port name and object name)

For example, consider this vision device:

The image shows two side-by-side screenshots of the Vision System object data in the VPM task tree. The left screenshot is for Impact Software versions before 9.0.0, and the right screenshot is for Impact Software version 9.0.0 and later. Both screenshots show the 'System' object expanded, displaying various properties and their values. The properties are: Name, Description, Type, Device Name, Device Comment, System Time, CPU Temperature, Available RAM, Model, Serial Number, Software Version, Kernel Version, Firmware Version, Board ID, and HTTP Port. The values for these properties are identical in both screenshots, except for the 'System Time' which is 8.664006E7 in the left screenshot and 1.589564E8 in the right screenshot. The 'Software Version' is 8.5.1 Build 7 in the left screenshot and 9.0.0 Build 179 in the right screenshot. The 'Kernel Version' is 8.5.1 Build 7 in the left screenshot and 9.0.0 Build 179 in the right screenshot. The 'Firmware Version' is 1 in the left screenshot and 11 in the right screenshot. The 'Board ID' is 255 in the left screenshot and 66 in the right screenshot. The 'HTTP Port' is 80 in both screenshots.

Property	Impact Software versions before 9.0.0	Impact Software version 9.0.0 and later
Name		
Description		
Type		
Device Name	IMPACT	IMPACT
Device Comment	IMPACT Device	IMPACT Device
System Time	8.664006E7	1.589564E8
CPU Temperature	53	53
Available RAM	162,307,600	162,284,496
Model	7	1
Serial Number	5	219
Software Version	8.5.1 Build 7	9.0.0 Build 179
Kernel Version	8.5.1 Build 7	9.0.0 Build 179
Firmware Version	1	11
Board ID	255	66
HTTP Port	80	80

For example, to get the Model value from the System (or General) system object, the URL will look like this:

`http://192.168.0.128/getData/Vision System.System:Model`

NOTE: The system object name “System” is used in this command even though the name is displayed as “General” in the VPM task tree in 9.0 and later.

You can put multiple commands in one line by separating them with a comma. For example, to get both the Model and the Serial Number in the same URL command, it will look like this:

```
http://192.168.0.128/getData/Vision System.System:Model,Vision
System.Serial Number
```

When the vision device receives a `getData` command, the integral web server returns the data values.

The following data is returned in the example.

```
<?xml version="1.0" encoding="utf-8" ?>
- <device>
  <String portname="Model">1</String>
  <String portname="Serial Number">219</String>
</ device >
```

Getting Task Tool Data

The command format to get data from a tool in a task is:

```
http://ip address/getData/program name.task name.tool name:port name
```

where

ip address is the IP Address of the vision device

getData is the command to get data from the device

program name is the vision program name

task name is the name of the task

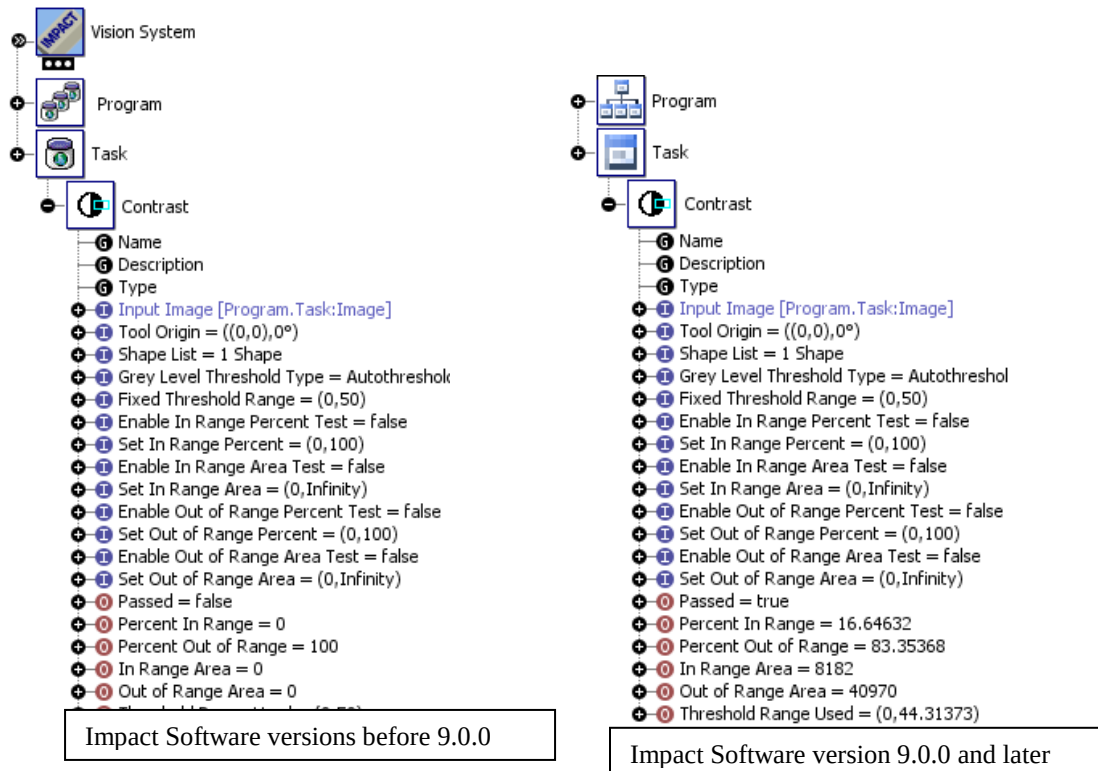
tool name is the name of the desired tool

port name is the name of the property (port) (Note that a colon separates the port name and tool name)

To get an image from the vision device, see Getting an image on page 8.

When the vision device receives a command, it returns the data values requested in the URL command.

For example, consider the following vision program:



the vision program name is *Program*

the task name is *Contrast Inspection Task*

the tool name is *Look for Contrast*

the desired property (port) name is *Percent In Range*

the device's IP Address is *192.168.0.128*

For example, to get the Percent In Range value from the Contrast tool, the URL will look like this:

`http://192.168.0.128/getData/Program.Task.Contrast:Percent In Range`

You can put multiple commands in one line by separating them with a comma. For example, to get both the Percent In Range and the In Range Area in the same URL command, it will look like this:

`http://192.168.0.128/getData/Program.Task.Contrast:Percent In Range,Program.Task.Contrast:In Range Area`

When the vision device receives a `getData` command, the integral web server returns the data values from the named port.

The following data is returned in the example.

```
<?xml version="1.0" encoding="utf-8" ?>
- < device >
- <Real portname="Percent In Range">
  <ver>2</ver>
  <val>16.646322</val>
  <min>-Inf</min>
  <max>Inf</max>
</Real>
- <Real portname="In Range Area">
  <ver>2</ver>
  <val>8182.000000</val>
  <min>-Inf</min>
  <max>Inf</max>
</Real>
</ device >
```

Getting an image

You can use the Image command to get an image from the camera and optionally scale its size. This command does not queue any tasks that are configured to run on an ImageInEvent. To use this command, the device must be offline. An image is returned as a PNG image MIME type. You cannot get both images and data with the same getData command.

NOTE: All commands and option names are case sensitive.

The command format to get an image is:

```
http://ip address/Image
```

where

ip address is the IP Address of the vision device

Image is the command to get an image

The command to get an image has optional parameters. The command format to scale the size of the image that is returned is:

```
http://ip address/Image/width=xxx,height=yyy
```

where

ip address is the IP Address of the vision device

Image is the command to get an image

width is the desired image width in pixels

height is the desired image height in pixels

If either the width or height parameter is omitted, the original camera image width or height is retained.

Scaling the image to a smaller size will speed up image acquisition when full resolution viewing is not required.

For example, to get an image that is not scaled (the image is full camera resolution), the URL would look like this:

```
http://192.168.0.128/Image
```

This example would return an image scaled to 100 pixels wide by 70 pixels high:

```
http://192.168.0.128/Image/width=100,height=70
```

This example would return an image scaled to 100 pixels wide by the original camera image height:

```
http://192.168.0.128/Image/width=100
```

This example would return an image scaled to the original camera image width by 1000 pixels high:

```
http://192.168.0.128/Image/height=1000
```

Setting Data

You can use the setData command to set integer, real, Boolean, and string values on the vision device. You can only set values on a tool or system object's input properties (ports).

The general command format to set data on the device has two forms, depending whether you are setting data in a vision system object or a task.

IMPORTANT NOTE: All commands and names (including Vision System, vision system object, program, task, and tool names) are case sensitive.

Setting Vision System Object Data

The command format to set data in a vision system object is:

```
http://ip address/setData/Vision System.vision system object name:port
name=(type)value
```

where

ip address is the IP Address of the vision device

setData is the command to set data in the device

Vision System is the two words Vision System (even though this is displayed as Impact in 9.0 and later)

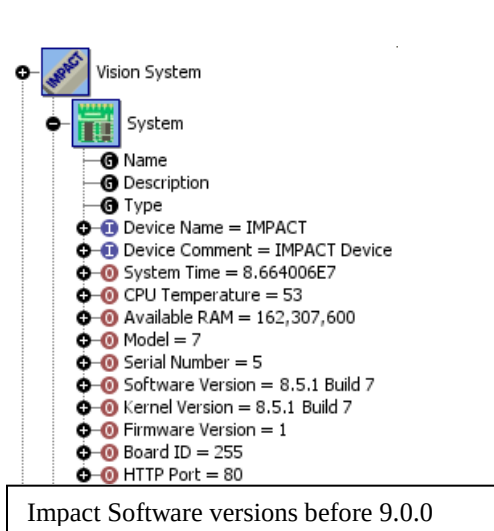
vision system object name is the name of the vision system object

port name is the name of the property (port) (Note that a colon separates the port name and object name)

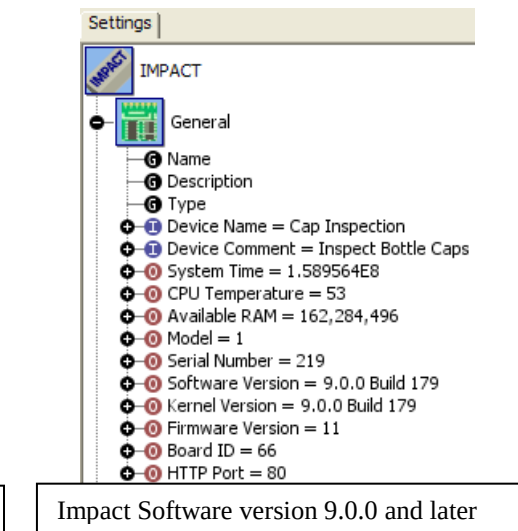
(type) is the port data type: integer, real, Boolean, or string

value is the desired setting value

For example, consider this vision device:



Impact Software versions before 9.0.0



Impact Software version 9.0.0 and later

To set the Device Name value in the System (or General) system object, the URL will look like this:

```
http://192.168.0.128/setData/Vision System.System:Device Name=(string)New Name
```

NOTE: The system object name "System" used is in this command even though the name is displayed as "General" in the VPM task tree in 9.0 and later.

You can put multiple commands in one line by separating them with a comma. For example, to set both the Device Name and Device Comment in the same URL command, it will look like this:

```
http://192.168.0.128/setData/Vision System.System:Device Name=(string)New
Name,Vision System.System:Device Comment=(string)New Comment
```

When the vision device receives a setData command, the integral web server response indicates whether the command was successful or not.

If the command was successful, the following data is returned:

```
<?xml version="1.0" encoding="utf-8" ?>
- < device >
  <SyntaxError>Ok</SyntaxError>
  <SetDataError>Ok</SetDataError>
</ device >
```

If the command was not successful, an error message is returned:

```
<?xml version="1.0" encoding="utf-8" ?>
- < device >
  <SyntaxError>Possible unknown port or type mismatch</SyntaxError>
  <SetDataError>Invalid Parameter</SetDataError>
</ device >
```

Setting Task Tool Data

The command format to set data in a task tool is:

```
http://ip address/setData/program name.task name.tool name:port name=(type)value
```

where

ip address is the IP Address of the vision device

setData is the command to set data on the device

program name is the vision program name

task name is the name of the task

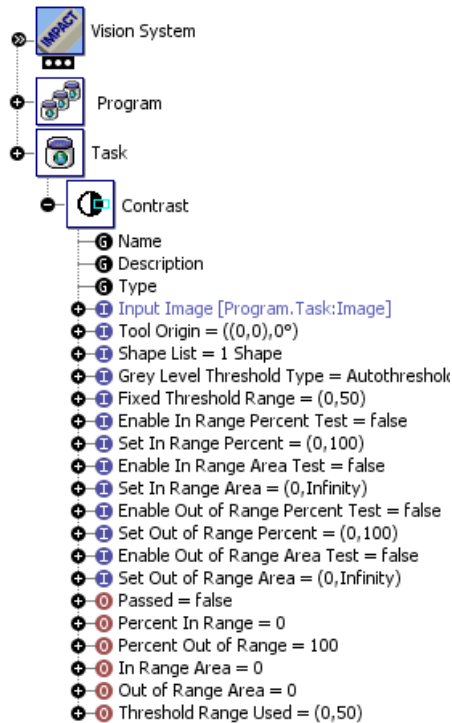
tool name is the name of the desired tool

port name is the name of the property (port) (Note that a colon separates the port name and tool name)

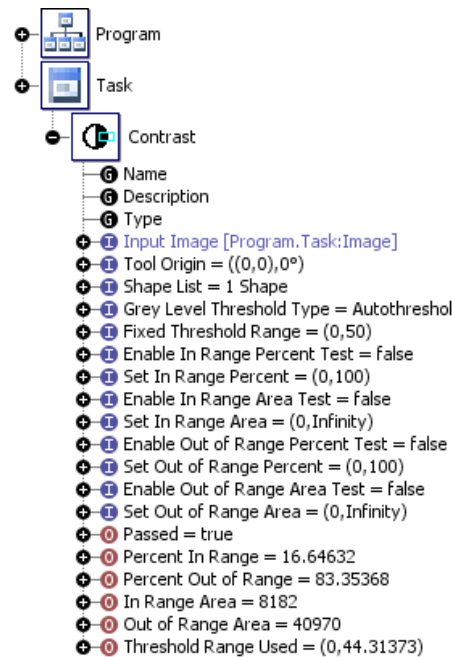
(type) is the port data type: integer, real, Boolean, or string

value is the desired setting value

For example, consider this vision program:



Impact Software versions before 9.0.0



Impact Software version 9.0.0 and later

To set the Contrast tool's Enable In Range Area Test value, the URL would look like this:

```
http://192.168.0.128/setData/Program.Task.Contrast:Enable In Range Area
Test=(boolean>false
```

You can put multiple commands in one line by separating them with a comma. For example, to set both the Enable In Range Area Test and Enable In Range Percent Test in the same URL command, it would look like this:

```
http://192.168.0.128/setData/Program.Task.Contrast:Enable In Range Area
Test=(boolean>false,Program.Task.Contrast:Enable In Range Percent
Test=(boolean>true
```

When the vision device receives a setData command, the integral web server response indicates whether the command was successful or not.

If the command was successful, the following data is returned:

```
<?xml version="1.0" encoding="utf-8" ?>
- < device >
  <SyntaxError>Ok</SyntaxError>
  <SetDataError>Ok</SetDataError>
</ device >
```

If the command was not successful, an error message is returned:

```
<?xml version="1.0" encoding="utf-8" ?>
- < device >
  <SyntaxError>Possible unknown port or type mismatch</SyntaxError>
  <SetDataError>Invalid Parameter</SetDataError>
</ device >
```

Triggering the camera

You can use the trigger command to trigger the camera which will queue any tasks that are configured to run on an ImageInEvent. The device must be online to accept the trigger command.

The general command format to trigger the camera on the device has an optional parameter, the trigger ID number. The trigger ID number can be used in the task for deciding whether the image requires special image processing.

The command format to trigger the camera is:

```
http://ip address/trigger
```

where

ip address is the IP Address of the vision device
trigger is the command to trigger the camera

The command format to trigger the camera and define the trigger ID number is:

```
http://ip address/trigger/id=n
```

where

ip address is the IP Address of the vision device
trigger is the command to trigger the camera
n is the desired trigger ID number

For example, to trigger the device at the following IP Address and assign an ID number, the URL would look like this:

```
http://192.168.0.128/trigger/id=1
```

Running a task or tool

You can use the Run command to run a task or tool on the vision device.

Task

The command format to run a task is:

```
http://ip address/run/program name.task name
```

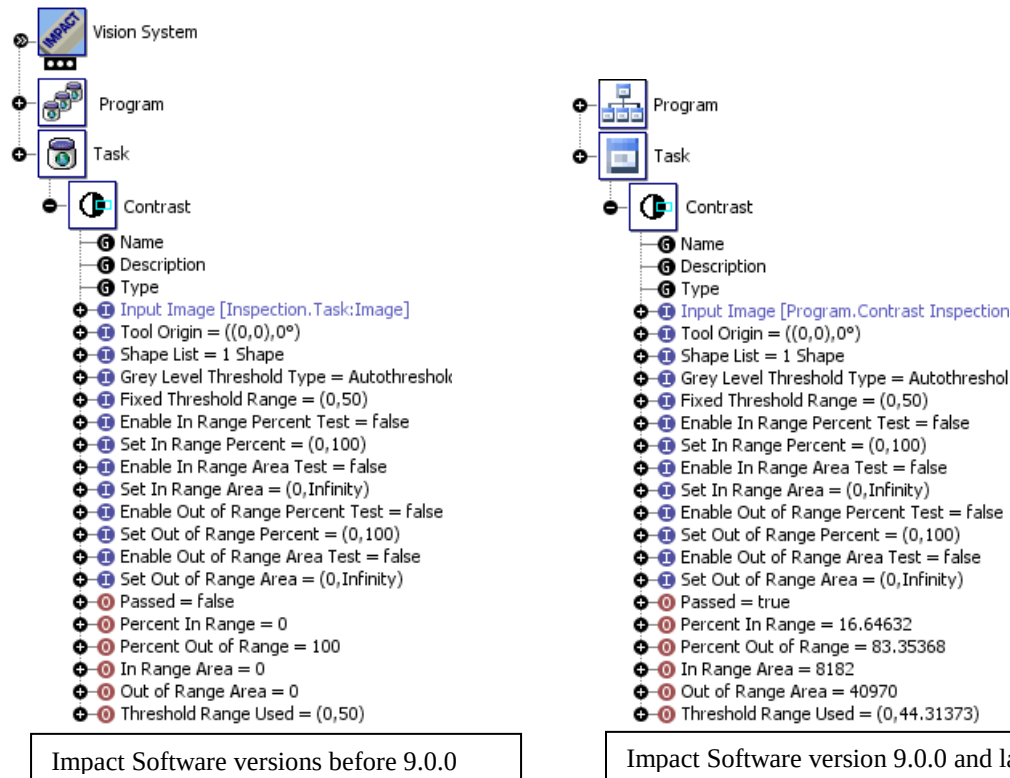
where

ip address is the IP Address of the vision device
run is the command to run the task

program name is the vision program name

task name is the name of the task

For example, consider this vision program:



For example, to run the task in the example, the URL will look like this:

`http://192.168.0.128/run/Program.Task`

You can put multiple commands in one line by separating them with a comma. For example, to run two tasks with the same URL command, it will look like this:

`http://192.168.0.128/run/Program.Task,Program.Task2`

Tool

The command format to run a tool is:

`http://ip address/run/program name.task name.tool name`

where

ip address is the IP Address of the vision device

run is the command to run the task

program name is the vision program name

task name is the name of the task

tool name is the name of the tool

For example, to run the Contrast tool in the example, the URL will look like this:

`http://192.168.0.128/run/Program.Task.Contrast`

You can put multiple commands in one line by separating them with a comma. For example, to run two tools with the same URL command, it would look like this:

`http://192.168.0.128/run/Program.Task.Contrast,Program.Task.Tool2`

Getting an image

You can use the Image command to get an image from the camera and optionally scale its size. This command does not queue any tasks that are configured to run on an ImageInEvent. To use this command, the device must be offline.

An image is returned as a PNG image MIME type. You cannot get both images and data with the same `getData` command.

NOTE: All commands and option names are case sensitive.

The command format to get an image is:

```
http://ip address/Image
```

where

ip address is the IP Address of the vision device

Image is the command to get an image

The command to get an image has optional parameters. The command format to scale the size of the image that is returned is:

```
http://ip address/Image/width=xxx,height=yyy
```

where

ip address is the IP Address of the vision device

Image is the command to get an image

width is the desired image width in pixels

height is the desired image height in pixels

If either the width or height parameter is omitted, the original camera image width or height is retained.

Scaling the image to a smaller size will speed up image acquisition when full resolution viewing is not required.

For example, to get an image that is not scaled (the image is full camera resolution), the URL would look like this:

```
http://192.168.0.128/Image
```

This example would return an image scaled to 100 pixels wide by 70 pixels high:

```
http://192.168.0.128/Image/width=100,height=70
```

This example would return an image scaled to 100 pixels wide by the original camera image height:

```
http://192.168.0.128/Image/width=100
```

This example would return an image scaled to the original camera image width by 1000 pixels high:

```
http://192.168.0.128/Image/height=1000
```