KNX IP BAOS 771 / 772 / 777

Web Services

Note:
The KNX IP BAOS 777 supports in addition RESTful web services which are specified in a separate document.

WEINZIERL ENGINEERING GmbH
DE-84508 Burgkirchen
E-Mail: info@weinzierl.de
Web: www.weinzierl.de

Index

History

Status of the document	Date	Editor
Creation (based on document BAOS_IP_Erweiterung_EN_2010_11_15_001.doc)	2010-11-30	Hl
Fixed Examples of GetDatapointValue: Format was missing in some examples	2010-12-10	Hl
Fixed: Example of GetServerItem	2011-06-15	Hl
Added: Infos for KNX IP BAOS 777	2015-07-30	Wz

1 Web-Services

To simplify the access on data-point level of the KNX IP BAOS (KNX application layer), web services are available as an alternative possibility. These web services are based on JSON (JavaScript Object Notation).

The parameter values in the URL are case sensitive.

1.1 GetServerItem

Request

/baos/GetServerItem?ItemStart=1&ItemCount=18

Response


```
{
	“Result”: true,
	“Service”	: “GetServerItem”,
	“Data”	:
	{
		“HardwareType”			: {0, 0, 197, 7, 0, 0},
		“HardwareVersion”		: 16,
		“FirmwareVersion”		: 16,
		“KnxManufacturerCodeDev”	: 197,
		“KnxManufacturerCodeApp”	: 197,
		“ApplicationId”			: 1796,
		“ApplicationVersion”		: 16,
		“SerialNumber”			: {0, 197, 7, 4, 0, 0},
		“TimeSinceReset”		: 28143,
		“BusConnectionState”		: 1,
		“MaximalBufferSize”		: 255,
		“LengthOfDescriptionString”	: 0,
		“Baudrate”			: 2,
		“CurrentBufferSize”		: 255,
		“ProgrammingMode”		: 0,
		“ProtocolVersion”		: 32,
		“IndicationSending”		: 1
		“ProtocolVersionWebServices”	: 32,
	},
}
```

1.2 SetServerItem

Request

/baos/SetServerItem?ProgrammingMode=0
/baos/SetServerItem?ProgrammingMode=1

/baos/SetServerItem?IndicationSending=0
/baos/SetServerItem?IndicationSending=1

/baos/SetServerItem?Baudrate=2

/baos/SetServerItem?BufferSize=255

Response

```
{
        “Result”: true,
        “Service”          : “SetServerItem”
}
```

1.3 GetDatapointDescription

Request

/baos/GetDatapointDescription?DatapointStart=1&DatapointCount=1

Response


```
{
	“Result”: true,
	“Service”	: “GetDatapointDescription”,
	“Data”	:
	[
		{
			“Datapoint”			: 1,
			“ValueType”			: 0,
			“ConfigurationFlags”	: 223,
			“DatapointType”			: 1

		}
	]
}
```

1.4 GetDescriptionString

Request

/baos/GetDescriptionString?DatapointStart=1&DatapointCount=1

Response


```
{
	“Result”: true,
	“Service”	: “GetDescriptionString”,
	“Data”	:
	[
		{
			“Datapoint”: 1, “Description”: “Switch”
		}
	]
}
```

1.5 GetDatapointValue

Request
/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Raw

Response

```
{
        "Result"        : true,

        "Service"       : "GetDatapointValue",
        "Data"  :
        [
                {
                        "Datapoint": 1,
                        "Format": "RAW",
                        "Length": 1,
                        "State": 0,
                        "Value": 0
                }
        ]
}
```

The desired format of the datapoint value is chosen by changing the parameter “Format”.

The following formats are supported:

Raw	RAW-Format: Data as array (byte-by-byte)
Default	Datapoint type (DPT), like the definition in Parameter

The following DPTs are supported:

DPT1	1 Bit (Boolean)
DPT2	2 Bit (Control)
DPT3	4 Bit (control Dimming, Control Blinds)
DPT4	8 Bit (Character Set)
DPT5	8 Bit Unsigned Value
DPT6	8 Bit Signed Value
DPT7	2 Byte Unsigned Value
DPT8	2 Byte Signed Value
DPT9	2 Byte Float Value
DPT10	3 Byte (Time)
DPT11	3 Byte (Date)
DPT12	4 Byte Unsigned Value
DPT13	4 Byte Signed Value
DPT14	4 Byte Float Value
DPT15	4 Byte (Access)
DPT16	14 Byte (String)
DPT17	1 Byte (Scene Number)
DPT18	1 Byte (Scene Control)

1.5.1 Example RAW-Format

Request
/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Raw

Response

```
{
	"Result" : true,
	"Service"	: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "RAW",
			"Length": 4,
			"State": 0,
			"Value": [0, 50, 200, 3]
		}
	]
}
```

1.5.2 Example DPT1-Format (Boolean)

Request
/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response

```
{
	"Result" : true,
	"Service"	: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "DPT1",
			"Length": 1,
			"State": 0,
			"Value": false
		}
	]
}
```

1.5.3 Example DPT2-Format (Control, 2 Bit)

Request

/baos/GetDatapointValue?DatapointStart=2&DatapointCount=1&Format=Default

Response


```
{
        "Result"      : true,
        "Service"     : "GetDatapointValue",
        "Data":
        [
          {
                "Datapoint": 2,
                "Format": "DPT2",
                "Length": 1,
                "State": 0,
                "Value":
                {
                        "Control": false,
                        "Code": false
                }
          }
        ]
}
```

1.5.4 Example DPT3-Format (Control Dimming, Control Blinds, 3 Byte)

Relative Dim: brighter, step 3

Request
/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response

```
{
	"Result" : true,
	"Service"	: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "DPT3",
			"Length": 1,
			"State": 0,
			"Value":
			{
				"Control": true,
				"StepCode": 3
			}
		}
	]
}
```

Relative dim: darker, step 2

Request
/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response

```
{
	"Result" : true,
	"Service"	: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "DPT3",
			"Length": 1,
			"State": 0,
			"Value":
			{
				"Control": false,
				"StepCode": 2
			}
		}
	]
}
```

Relative Dim: stop

Request

/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format= Default

Response


```
{
	"Result" : true,
	"Service"		: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "DPT3",
			"Length": 1,
			"State": 0,
			"Value":
			{
				"Control": false,
				"StepCode": 0
			}
		}
	]
}
```

1.5.5 Example DPT4-Format (Character Set, 1 Byte)

Request

/baos/GetDatapointValue?DatapointStart=4&DatapointCount=1&Format=Default

Response


```
{
        "Result"      : true,
        "Service"     : "GetDatapointValue",
        "Data"        :
          [
                {
                        "Datapoint": 4,
                        "Format": "DPT4",
                        "Length": 1,
                        "State": 0,
                        "Value": "A"
                }
          ]
}
```


1.5.6 Example DPT5-Format (Unsigned Value, 1 Byte)

Request

/baos/GetDatapointValue?DatapointStart=5&DatapointCount=1&Format=Default

Response


```
{
        "Result"      : true,
        "Service"     : "GetDatapointValue",
        "Data"        :
          [
                {
                        "Datapoint": 5,
                        "Format": "DPT5",
                        "Length": 1,
                        "State": 0,
                        "Value": 20
                }
          ]
}
```

1.5.7 Example DPT6-Format (Signed Value, 1 Byte)

Request

/baos/GetDatapointValue?DatapointStart=6&DatapointCount=1&Format=Default

Response


```
{
	"Result"	: true,
	"Service"	: "GetDatapointValue",
	"Data"		:
	  [
		{
			"Datapoint": 6,
			"Format": "DPT6",
			"Length": 1,
			"State": 0,
			"Value": -123
		}
	  ]
}
```


1.5.8 Example DPT7-Format (Unsigned Value, 2 Byte)

Request

/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
	"Result" : true,
	"Service"		: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "DPT7",
			"Length": 2,
			"State": 0,
			"Value": 8415
		}
	]
}
```

1.5.9 Example DPT8-Format (Signed Value, 2 Byte)

Request

/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
	"Result" : true,
	"Service"	: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "DPT8",
			"Length": 2,
			"State": 0,
			"Value": -837
		}
	]
}
```


1.5.10 Example DPT9-Format (Float, 2 Byte)

Request

/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
	"Result" : true,
	"Service"	: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "DPT9",
			"Length": 2,
			"State": 0,
			"Value": 22.5
		}
	]
}
```

1.5.11 Example DPT10-Format (Time, 3 Byte)

Request

/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
	"Result" : true,
	"Service"	: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "DPT10",
			"Length": 3,
			"State": 0,
			"Value":
			{
				"Weekday": "Monday",
				"Hour": 10,
				"Minute": 45,
				"Second": 30
			}
		}
	]
}
```


1.5.12 Example DPT11-Format (Date, 3 Bytes)

Request

/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
	"Result" : true,
	"Service"	: "GetDatapointValue",
	"Data"	:
	[
		{
			"Datapoint": 1,
			"Format": "DPT11",
			"Length": 3,
			"State": 0,
			"Value":
			{
				"Day": 31,
				"Month": 12,
				"Year": 2009
			}
		}
	]
}
```

1.5.13 Example DPT12-Format (Unsigned Value, 4 Byte)

Request

/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
        "Result"      : true,
        "Service"     : "GetDatapointValue",
        "Data"        :
          [
                {
                        "Datapoint": 12,
                        "Length": 3,
                        "State": 0,
                        "Value": 12345
                }
          ]
}
```


1.5.14 Example DPT13-Format (Signed Value, 4 Byte)

Request

/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
        "Result"      : true,
        "Service"     : "GetDatapointValue",
        "Data"        :
          [
                {
                        "Datapoint": 13,
                        "Length": 4,
                        "State": 0,
                        "Value": -12345
                }
          ]
}
```

1.5.15 Example DPT14-Format (Float. 4 Byte)

Request
/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
	"Result"	: true,
	"Service"	: "GetDatapointValue",
	"Data"		:
	  [
		{
			"Datapoint": 14,
			"Length": 4,
			"State": 0,
			"Value": 22.157
		}
	  ]
}
```


1.5.16 Example DPT15-Format (Access, 14 Byte)

Request
/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
	"Result" : true,
	"Service"		: "GetDatapointValue",
	"Data"			:
	[
		{
			"Datapoint": 1,
			"Format": "DPT15",
			"Length": 4,
			"State": 0,
			"Value":
			{
				"Code": 45615,
				"Index": 5,
				"FlagError": true,
				"FlagPermission": false,
				"FlagReadDirection": false,
				"FlagEncrypted": true
			}
		}
	]
}
```

1.5.17 Example DPT16-Format (String, 14 Byte)

Request
/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response

```
{
	"Result" : true,
	"Service"	: "GetDatapointValue",
	"Data"		:
	[
		{
			"Datapoint": 1,
			"Format": "DPT16",
			"Length": 14,
			"State": 0,
			"Value": "Test"
		}
	]
}
```


1.5.18 Example DPT17-Format (Scene Number, 1 Byte)

Request
/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response

```
{
	"Result"	: true,
	"Service"	: "GetDatapointValue",
	"Data"		:
	   [
			{
				"Datapoint": 17,
				"Length": 14,
				"State": 0,
				"Value": 0
			}
	   ]
}
```

1.5.19 Example DPT18-Format (Scene Control, 1 Byte)

Request

/baos/GetDatapointValue?DatapointStart=1&DatapointCount=1&Format=Default

Response


```
{
        "Result"      : true,
        "Service"     : "GetDatapointValue",
        "Data"        :
          [
                {
                        "Datapoint": 18,
                        "Length": 1,
                        "State": 0,
                        "Value":
                        {
                                "Control": false,
                                "Scene": 0
                        }
                }
          ]
}
```

1.6 SetDatapointValue

Request
/baos/SetDatapointValue?Datapoint=1&Format=RAW&Length=1&Value=0
/baos/SetDatapointValue?Datapoint=1&Format=RAW&Command=SetSendVal&Length=1&Value=0

Response
```
{
        “Result”: true
        “Service”        : “SetDatapointValue”
}
```

This service is limited to one data point.
If the parameter “Command” is not stated in the request, “Set new value and send on bus” is executed.

Commands:

SetVal	Set new value
SendVal	Send value on bus
SetSendVal	Set new value and send on bus
ReadVal	Read new value via bus
ClrState	Clear datapoint transmission state

1.6.1 Example RAW-Format (hexadecimal)

RAW-Format (hexadecimal) length 1 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=Raw&Length=1&Value=0xA2

Response
```
{
	“Result”: true
	“Service”	: “SetDatapointValue”
}
```

RAW-Format (hexadecimal) length 2 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=Raw&Length=2&Value=0x0055

Response
```
{
	“Result”: true
	“Service”	: “SetDatapointValue”
}
```

RAW-Format (hexadecimal) length 4 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=Raw&Length=4&Value=0x12345678

Response
```
{
	“Result”: true
	“Service”	: “SetDatapointValue”
}
```

RAW-Format (hexadecimal) length 14 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=Raw&Length=14&Value=0x112233445566778899AABBCCDDEE

Response
```
{
	“Result”: true
	“Service”	: “SetDatapointValue”
}
```

1.6.2 Example RAW-Format (decimal)

RAW-Format (hexadecimal) length 1 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=Raw&Length=1&Value=25

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

RAW-Format (decimal) length 2 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=Raw&Length=2&Value=42910

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

RAW-Format (decimal) length 4 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=Raw&Length=4&Value=1575185641

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.3 Example DPT1-Format (Boolean, 1Bit)

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT1&Length=1&Value=true

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT1&Length=1&Value=false

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.4 Example DPT2-Format (Control, 2Bit)

Request

/baos/SetDatapointValue?Datapoint=1&Format=DPT2&Length=1&Value=true&Control=true

Response

```
{
        “Result”: true
        “Service”        : “SetDatapointValue”
}
```

Request

/baos/SetDatapointValue?Datapoint=1&Format=DPT2&Length=1&Value=false&Control=false

Response

```
{
        “Result”: true
        “Service”        : “SetDatapointValue”
}
```

1.6.5 Example DPT3-Format (Control Dimming, Control Blinds, 4 Bit)

Relative Dim: lighter, step 3

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT3&Length=1&Control=true&StepCode=3

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

Relative ´Dim: darker, step 1

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT3&Length=1&Control=false&StepCode=1

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

Relative Dim: stop

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT3&Length=1&Control=false&StepCode=0

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.6 Example DPT4-Format (Character Set, 1 Byte)

Request

/baos/SetDatapointValue?Datapoint=1&Format=DPT4&Length=1&Value=T

Response

{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.7 Example DPT5-Format (Unsigned Value, 1 Byte)

UINT-Format length 1 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT5&Length=1&Value=123

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.8 Example DPT6-Format (Signed Value, 1 Byte)

INT-Format length 1 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT6&Length=1&Value=-25

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.9 Example DPT7-Format (Unsigned Value, 2 Byte)

UINT-Format length 2 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT7&Length=2&Value=45201

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.10 Example DPT8-Format (Signed Value, 2 Byte)

INT-Format length 2 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT8&Length=2&Value=-25301

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.11 Example DPT9-Format (Float, 2 Byte)

Float-Format length 2 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT9&Length=2&Value=23.1

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.12 Example DPT10-Format (Time, 3 Byte)

Request

/baos/SetDatapointValue?Datapoint=1&Format=DPT10&Length=3&Weekday=Monday&Hour=10&Minute=45&Second=30

Response

```
{
        “Result”: true
        “Service”        : “SetDatapointValue”
}
```

The following values are possible:

Weekday:	Monday, Tuesday, Wednesday, Thursday, Friday, Saturday & Sunday
Hour:	0 … 23
Minute:	0 … 59
Second:	0 … 59

1.6.13 Example DPT11-Format (Date, 3 Byte)

Request

/baos/SetDatapointValue?Datapoint=1&Format=DPT11&Length=3&Day=31&Month=12&Year=2009

Response

```
{
        “Result”: true
        “Service”        : “SetDatapointValue”
}
```

1.6.14 Example DPT12-Format (Unsigned Value, 4 Byte)

UINT-Format length 4 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT12&Length=4&Value=45201

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.15 Example DPT13-Format (Signed Value, 4 Byte)

INT-Format length 4 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT13&Length=4&Value=-25301

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.16 Example DPT14-Format (Float, 4 Byte)

Float-Format length 4 Byte:

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT14&Length=4&Value=23.1

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.17 Example DPT15-Format (Access, 4 Byte)

Request

/baos/SetDatapointValue?Datapoint=1&Format=DPT15&Length=4&Code=45615&Index=5&FlagError=true&FlagPermission=false&FlagReadDirection=false&FlagEncrypted=true

Response

{
“Result”: true
“Service” : “SetDatapointValue”
}

1.6.18 Example DPT16-Format (String, 14 Byte)

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT16&Length=14&Value=Test

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

Using the parameter „Value“ a string with a maximum length of 14 bytes can be given to the BAOS.

1.6.19 Example DPT17-Format (Scene Number, 1 Byte)

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT17&Length=1&Scene=60

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

Scene numbers from 0 to 63 can be used.

1.6.20 Example DPT18-Format (Scene Control, 1 Byte)

Request
/baos/SetDatapointValue?Datapoint=1&Format=DPT18&Length=1&Scene=60&Control=true

Response
{
“Result”: true
“Service” : “SetDatapointValue”
}

Scene numbers from 0 to 63 can be used.

1.7 GetParameterByte

Request

/baos/GetParameterByte?ByteStart=1&ByteCount=15

Response

```
{
        “Result”: true,
        “Service”          : “GetParameterByte”,
        “Data”   :
        {
                255, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 0, 0, 0
        }
}
```

1.8 StartIndicationSession

Request
/baos/StartIndicationSession

Response

```
{
        “Result”: true,
        “Service”          : “StartIndicationSession”,
        “Data”   :
        {
                “SessionId”: 1
        }
}
```

To draw a link between these two connections (Requ-Resp and Ind.), an unique session ID is used.
This Session-ID is generated by the BAOS and has of length of 2 bytes.
Session IDs in the range of 1 to 65535 are assigned.

An inactive indication session will be terminated after 30 seconds.

1.9 StopIndicationSession

Request

/baos/StopIndicationSession&SessionId=1

Response

```
{
        “Result”: true,
        “Service”          : “StopIndicationSession”
}
```

1.10 GetIndication

This service is used to request Indications. A feature of this service is that it remains open for a certain time (Timeout, in seconds). This means, the service waits for the Indication after a Request is posted.

The timeout is limited to 30 seconds.

If there is no Indication within the defined period of time, a response containing the value FALSE in the Parameter “Result” is posted.

The desired format of the datapoint value is chosen by changing the parameter “Format”.

The following formats are supported:
Raw RAW-Format: Data as array (byte-by-byte)
Default Datapoint type (DPT), like the definition in Parameter

If no format is set, the RAW-Format will be used for datapoint values.

This service is used to get indications of server items and of datapoint values.

Example for receiving a bus connection change:

Request
/baos/GetIndication?SessionId=1&Timeout=30&Format=RAW
or
/baos/GetIndication?SessionId=1&Timeout=30

Response
```
{
        "Result"    : true,
        "Service"   : "GetIndication",
        "Type"      : "ServerItem"
        "Data"      :
          {
                    "BusConnectionState": 1
          }
}
```

The parameter “Format” has no influence on a server item indication.

Example for receiving datapoint value indications:

Request

/baos/GetIndication?SessionId=1&Timeout=30&Format=RAW

or

/baos/GetIndication?SessionId=1&Timeout=30

Response


```
{
        "Result"      : true,
        "Service"     : "GetIndication",
        "Type"        : "DatapointValue"
        "Data"        :
           [
                {
                        "Datapoint": 1,
                        "Length": 1,
                        "State": 0,
                        "Value": [0]
                }
           ]
}
```


Request

/baos/GetIndication?SessionId=1&Timeout=30&Format=Default

Response


```
{
        “Result”: true,
        “Service”          : “GetIndication”,
        "Type"        : "DatapointValue"
        “Data”   :
        [
                {
                        “Datapoint”: 1, “Format”: DPT1, “Length”: 1, “State”: 0, “Value”: false
                }
        ]
}
```

1.11 Error messages

In case of an error the answer with an error description is returned.

Response

```
{
        “Result”: false,
        “Service”        : “GetParameterByte”,
        “Error”  :”ErrorMessage”,
}
```

1.11.1 Unsupported Service

“Error” :”UnsupportedService”

This error message is returned, if an unsupported web service is sent to the KNX IP BAOS device.

Example:
/baos/IllegalService

1.11.2 Invalid Parameter

“Error” :”InvalidParam”

This error message is returned, if an invalid parameter is sent to the KNX IP BAOS device.

Example:
/baos/GetDatapointValue?DatapointStart=-1000&DatapointCount=1&Format=Raw

1.11.3 No Data Available

“Error” :”NoDataAvailable”

This error is returned, if no data is available, e.g. when asking for a description string and no strings are set via ETS.

1.11.4 Unknown Error

“Error” :”UnknownError”

This error is returned, if no specific error code is available.

1.11.5 Internal Error

“Error” :”InternalError”

This is an unspecified internal error of the KNX IP BAOS device.

1.11.6 Item Not Supported

“Error” :”ItemNotSupported”

This error is returned, if the requested server item is not available.

1.11.7 Buffer Too Small

“Error” :”BufferTooSmall”

This error is returned, if the internal buffer of the object server is not sufficient.

1.11.8 Item Not Writable

“Error” :”ItemNotWritable”

This error is returned, if a server item, which is not write enabled, shall be written.

1.11.9 Bad Service Parameter

“Error” :”BadServiceParameter”

This error is returned, if an object server parameter is out of range

1.11.10 Bad Object ID

“Error” :”BadObjectId”

This error is returned, if a non-existing object shall be set.

1.11.11 Bad Object Command

“Error” :”BadObjectCommand”

This error is returned, if an object shall be set with an invalid command.

1.11.12 Bad Length

“Error” :”BadLength”

This error is returned, if the length in the request does not match with the object length.

1.11.13 No Free Session

“Error” :”NoFreeSession”

This error is returned, if no indication session could be started, as no free sessions are available.

1.11.14 Session ID Not Found

“Error” :”SessionIdNotFound”

This error is returned, if the indication session could not be stopped, as the session ID could not be found.

1.11.15 Link To Session Failed

“Error” :”LinkToSessionFailed”

This error is returned, if an indication could not be retrieved, as the link to a session, which should exist, failed.

1.11.16 Release Session Failed

“Error” :”ReleaseSessionFailed”

This error is returned, if the releasing of an indication session failed (e.g. at timeout).

1.11.17 Indication Timeout

“Error” :”IndTimeout”

This error is returned, if a timeout of an existing indication session occurred.

1.11.18 Indication Stopped

“Error” :”IndStopped”

This error is returned, if an existing indication session has been stopped.

Example:
/baos/StopIndicationSession&SessionId=1

2 API

To simplify the use of the web services provided by the KNX IP BOAS, an API (application programming interface) for Javascript is available.
This API is based on jQuery JavaScript Library (http://jquery.com/) and jQuery JSONP Core Plugin (http://code.google.com/p/jquery-jsonp/).

The API provides the following functionality:

- API_SetIpAddress(strIpAddr)
- API_SetCallbackRespRcvd(strCallback)
- API_SetCallbackIndicationUpdate(strCallback)
- API_SetCallbackInvalidSettings(strCallback)
- API_SetCallbackTransmitError(strCallback)
- API_GetServerItem(strStartItem, strItemCount)
- API_GetDatapointDescription(strDatapointStart, strDatapointCount)
- API_GetDescriptionString(strDatapointStart, strDatapointCount)
- API_GetDatapointValue(strDatapointStart, strDatapointCount, strFormat)
- API_SetDatapointValue(strDatapoint, strFormat, strCommand, strLength, strValue)
- API_GetParamByte(strByteStart, strByteCount)
- API_IsIpAddrValid(strIpAddrToValidate)
- API_StartIndicationListener(strFormat)
- API_StopIndicationListener()

2.1 API_SetIpAddress(strIpAddr)

This function gives the IP address information to the API.

2.2 API_SetCallbackRespRcvd(strCallback)

This function sets the callback handler for processing the responses.

2.3 API_SetCallbackIndicationUpdate(strCallback)

This function sets the callback handler for processing the indication updates.

2.4 API_SetCallbackInvalidSettings(strCallback)

This function sets the callback handler for processing invalid settings. The callback function is called, if an API with invalid parameters is called.

2.5 API_SetCallbackTransmitError(strCallback)

This function sets the callback handler for processing transmission errors.

2.6 API_GetServerItem(strStartItem, strItemCount)

This API is called by the client to get one or more server items (properties). The start item (strStartItem) and the item count (strItemCount) have to be passed to the API.

2.7 API_GetDatapointDescription(strDatapointStart, strDatapointCount)

This API is called by the client to get one or more description(s) of the datapoint(s). The start datapoint (strDatapointStart) and the datapoint count (strDatapointCount) have to be passed to the API.

2.8 API_GetDescriptionString(strDatapointStart, strDatapointCount)

This API is called by the client to get the human-readable description string(s) of the datapoint(s). The start datapoint (strDatapointStart) and the datapoint count (strDatapointCount) have to be passed to the API.

2.9 API_GetDatapointValue(strDatapointStart, strDatapointCount, strFormat)

This API is called by the client to get the value(s) of the datapoint(s). The start datapoint (strDatapointStart), the datapoint count (strDatapointCount) and the format of the value (strFormat) have to be passed to the API.

2.10 API_SetDatapointValue(strDatapoint, strFormat, strCommand, strLength, strValue)

This API is called by the client to set the new value(s) of the datapoint(s) or to request/transmit the new value on the bus. The datapoint to set (strDatapoint), the format of the value (strFormat), the command (strCommand), the length of the datapoint (strLength) and its value (strValue) have to be passed to the API.

2.11 API_GetParamByte(strByteStart, strByteCount)

This API is called by the client to get the parameter byte(s). A parameter is a free-defined variable of the 8-bits length, which can be set and programmed by the Engineering Tool Software (ETS). The start index (strByteStart) and the parameter byte count (strByteCount) have to be passed to the API.

2.12 API_IsIpAddrValid(strIpAddrToValidate)

This API checks, if a given IP address (strIpAddrToValidate) is valid.
Return:
TRUE, if IP address is valid.
FALSE, in case of error

2.13 API_StartIndicationListener(strFormat)

This API is called to start the indication listener. The format of the datapoint value has to be passed to the API.
TRUE, if indication listener could be started successfully
FALSE, in case of error

2.14 API_StopIndicationListener()

This API is called to stop the indication listener.
TRUE, if indication listener could be stopped successfully
FALSE, in case of error