New Features in ibaPDA v8.13.0

Date: 2026-02-11

Table of contents

1 General 3

2 Virtual Functions 3

2.1 Text functions 3

2.1.1 ConvertToText 3

2.1.2 ConvertFromText 4

2.2 MMinDynamic and MMaxDynamic 5

3 Kafka data store: support for AVRO (grouped) data format 6

4 New e-mail fields related to clients 8

5 User management improvements 9

5.1 Support for Active Directory display names 9

5.2 Improved help texts for user rights 10

6 Locking layouts 11

7 TCP port used by S7 connections 13

1 General

ibaM-DAQ requires at least firmware package v01.06.001 or later.

ibaM-DAQ and ibaM-COM require both ibaPDA server and client to be v8.13.0 or later.

MindSphere data stores are no longer supported.

2 Virtual Functions

2.1 Text functions

In ibaPDA the virtual functions ConvertToText and ConvertFromText are used to convert between data signals and text signals.
In the newest release, these functions have been extended with more functionality, specifically handling different number bases. This includes binary, octal and hexadecimal numbers.

2.1.1 ConvertToText

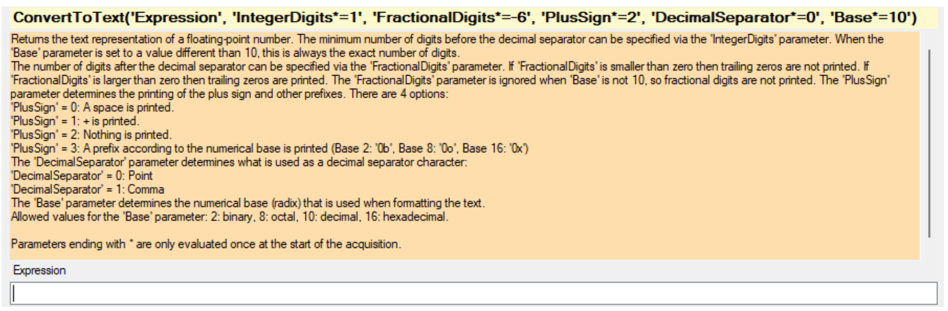
The ConvertToText function has received an additional parameter “Base” that determines the output numerical base. Possible values are 2 for binary, 8 for octal, 10 for the usual decimal representation and 16 for hexadecimal.

When the “Base” parameter is set to a value different than 10, it interacts with other parameters.

The “IntegerDigits” parameter also shortens numbers when the base is different than 10. The behaviour in base 10 was and is to only pad the number to a minimum size.

The “FractionalDigits” and “DecimalSeparator” parameters are ignored entirely when a different base than 10 is used. The behaviour here is to round the number and therefore remove the fractional part. Rounding is done in the same way as setting “FractionalDigits” to zero, rounding away from zero.

The “PlusSign” parameter has a new accepted value 3, which prepends a prefix according to the base. The most common use for this is adding “0x” to hex numbers, but binary and octal numbers also get their own prefix, “0b” and “0o” respectively.



ConvertToText('Expression', 'IntegerDigits*=1', 'FractionalDigits*=-6', 'PlusSign*=2', 'DecimalSeparator*=0', 'Base*=10')

Returns the text representation of a floating-point number. The minimum number of digits before the decimal separator can be specified via the 'IntegerDigits' parameter. When the 'Base' parameter is set to a value different than 10, this is always the exact number of digits.
The number of digits after the decimal separator can be specified via the 'FractionalDigits' parameter. If 'FractionalDigits' is smaller than zero then trailing zeros are not printed. If 'FractionalDigits' is larger than zero then trailing zeros are printed. The 'FractionalDigits' parameter is ignored when 'Base' is not 10, so fractional digits are not printed. The 'PlusSign' parameter determines the printing of the plus sign and other prefixes. There are 4 options:
'PlusSign' = 0: A space is printed.
'PlusSign' = 1: + is printed.
'PlusSign' = 2: Nothing is printed.
'PlusSign' = 3: A prefix according to the numerical base is printed (Base 2: '0b', Base 8: '0o', Base 16: '0x')
The 'DecimalSeparator' parameter determines what is used as a decimal separator character:
'DecimalSeparator' = 0: Point
'DecimalSeparator' = 1: Comma
The 'Base' parameter determines the numerical base (radix) that is used when formatting the text.
Allowed values for the 'Base' parameter: 2: binary, 8: octal, 10: decimal, 16: hexadecimal.

Parameters ending with * are only evaluated once at the start of the acquisition.

Expression

When converting negative numbers to text, two’s complement representation is used. This uses 64 bits of precision, so the resulting strings become very large. To limit the length of the string, the “IntegerDigits” parameter is used. It is important to note, that excess digits are simply cut off, which can lead to negative numbers “looping”, therefore becoming positive again, and vice-versa.

ConvertToText with a “Base” value different than 10 internally always works on 64-bit signed integers, so signal values larger than 2^{63} – 1 are not correctly converted.

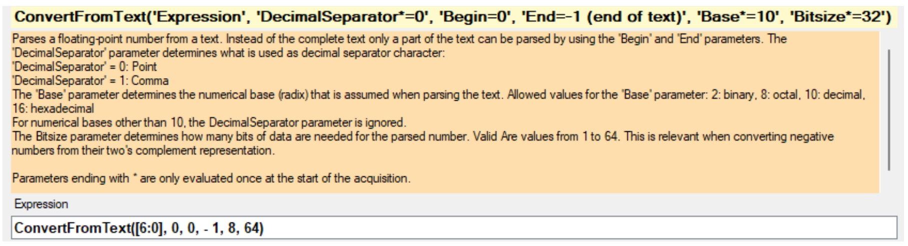
2.1.2 ConvertFromText

The ConvertFromText function has received two additional parameters, “Base” and “Bitsize”.

The “Base” parameter determines the numerical base expected in the text input. Possible values are 2 for binary, 8 for octal, 10 for the usual decimal representation and 16 for hexadecimal.

The “Bitsize” parameter is only relevant when the “Base” parameter is set to something different than 10. It determines how many bits the parsed number should have. This is necessary for correctly parsing numbers that are represented in two’s complement.

For example, parsing the number 0xFFFF results in -1 when using “Bitsize=16”, but it results in 65535 when a larger Bitsize is used.



ConvertFromText('Expression', 'DecimalSeparator*=0', 'Begin=0', 'End=-1 (end of text)', 'Base*=10', 'Bitsize*=32')

Parses a floating-point number from a text. Instead of the complete text only a part of the text can be parsed by using the 'Begin' and 'End' parameters. The 'DecimalSeparator' parameter determines what is used as decimal separator character:
'DecimalSeparator' = 0: Point
'DecimalSeparator' = 1: Comma
The 'Base' parameter determines the numerical base (radix) that is assumed when parsing the text. Allowed values for the 'Base' parameter: 2: binary, 8: octal, 10: decimal, 16: hexadecimal
For numerical bases other than 10, the DecimalSeparator parameter is ignored.
The Bitsize parameter determines how many bits of data are needed for the parsed number. Valid Are values from 1 to 64. This is relevant when converting negative numbers from their two's complement representation.

Parameters ending with * are only evaluated once at the start of the acquisition.

Expression

```
ConvertFromText([6:0], 0, 0, - 1, 8, 64)
```

ConvertFromText with a “Base” value different than 10 internally always works on 64-bit signed integers, so signal values larger than 2^{63} – 1 are not correctly converted.

If the parameter “Begin” is less than 0 it will be internally converted to 0.

2.2 MMinDynamic and MMaxDynamic

There exist already multiple functions to find the minimum and maximum values of signals.

- **Max**('Expression', 'Reset=0'):

  This calculates the maximum of a signal since the last reset.

- **MaxInTime**('Expression', 'WindowInterval', 'Reset=0')

  This calculates the maximum of a signal in an interval. Once the interval has passed a new maximum is calculated. The calculation will then start from scratch with a new interval. The length of the interval is fixed and can only be changed on a reset.

- **MMax**('Expression', 'WindowInterval', 'UpdateInterval=timebase', 'Reset=0')

  This calculates the maximum of a signal in an interval that is moved every sample. This means that a new maximum can be produced every sample. The length of the interval is fixed and can only be changed on a reset. The first maximum is returned once the interval has completely passed.

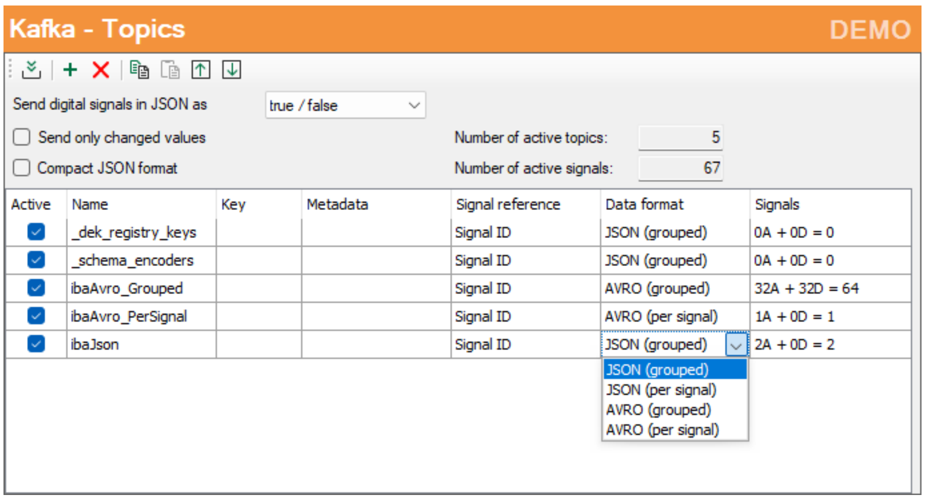
- **MMaxDynamic**('Expression', 'WindowInterval', 'MaxWindowInterval*')

  This calculates the maximum of a signal in an interval that is moved every sample. The length of the interval can also change every sample. The interval length is limited to the configured maximum interval. A new maximum is produced every sample also when the interval hasn’t completely passed.

The MMaxDynamic and MMinDynamic functions could be used to calculate the minimum and maximum value during the last meter when the interval is calculated based on a speed signal.

3 Kafka data store: support for AVRO (grouped) data format

In the Kafka cluster timebased data store, it is now possible to select AVRO (grouped) as a data format for encoding the data that is sent to the configured Kafka cluster.



Send digital signals in JSON as: true / false

- [ ] Send only changed values
- [ ] Compact JSON format

Number of active topics: 5

Number of active signals: 67

Active	Name	Key	Metadata	Signal reference	Data format	Signals
✓	_dek_registry_keys			Signal ID	JSON (grouped)	0A + 0D = 0
✓	_schema_encoders			Signal ID	JSON (grouped)	0A + 0D = 0
✓	ibaAvro_Grouped			Signal ID	AVRO (grouped)	32A + 32D = 64
✓	ibaAvro_PerSignal			Signal ID	AVRO (per signal)	1A + 0D = 1
✓	ibaJson			Signal ID	JSON (grouped)	2A + 0D = 2

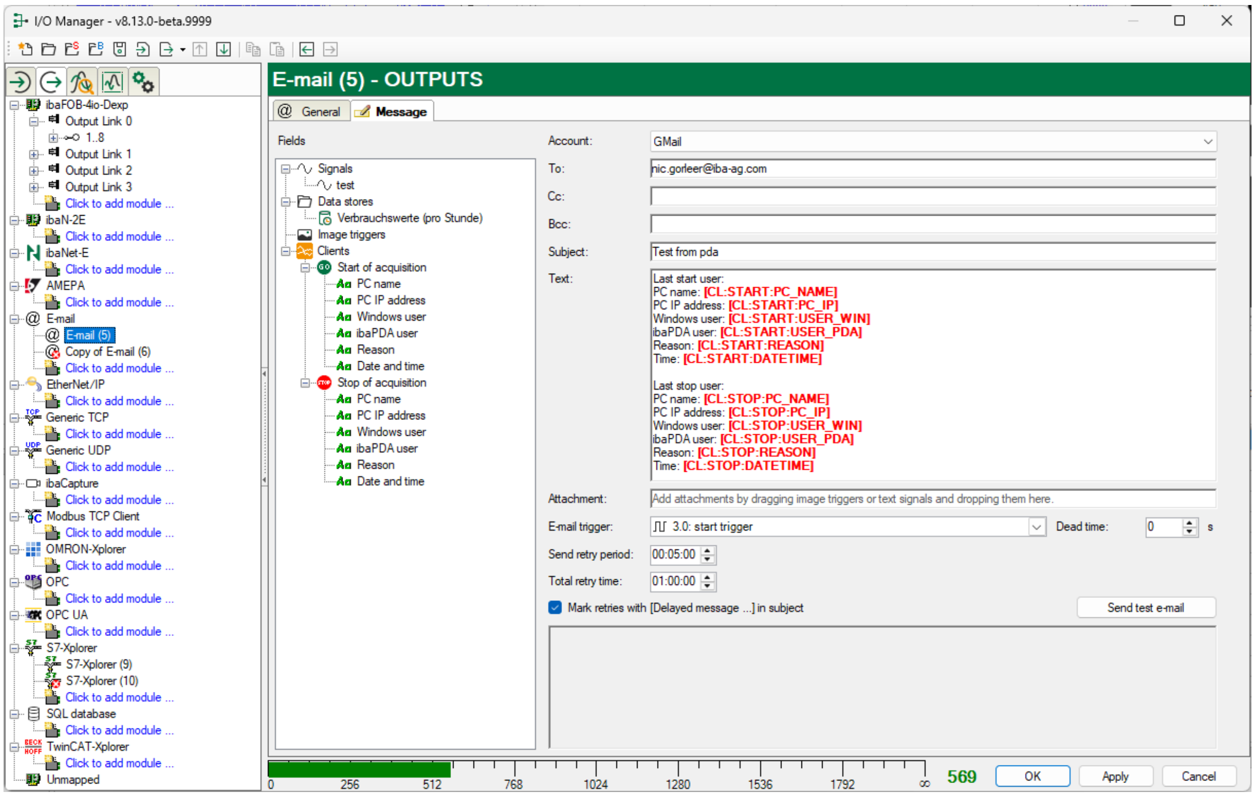
The following AVRO schema is used:

```
{
"namespace": "de.iba",
"type": "record",
"name": "PdaGroupedRecord",
"fields":
    [
        {
            "name": "Signal",
            "type": { "type": "array", "items": "string" }
        },
        {
            "name": "ID",
            "type": [ "null", { "type": "array", "items": "string" } ]
        },
        {
            "name": "Name",
            "type": [ "null", { "type": "array", "items": "string" } ]
        },


        {
            "name": "Unit",
            "type": [ "null", { "type": "array", "items": "string" } ]
        },
        {
            "name": "Comment1",
            "type": [ "null", { "type": "array", "items": "string" } ]
        },
        {
            "name": "Comment2",
            "type": [ "null", { "type": "array", "items": "string" } ]
        },
        {
            "name": "Timestamp",
            "type": [ "null", { "type": "long", "logicalType": "timestamp-micros" } ]
        },
        {
            "name": "Identifier",
            "type": [ "null", "string" ]
        },
        {
            "name": "Value",
            "type": { "type": "array", "items": { "type": ["null", "boolean", "bytes",
"double", "float", "int", "long", "string"] } }
        }
    ]
}
```

4 New e-mail fields related to clients

The e-mail module can send e-mails. The e-mail message can contain different fields to include dynamic content. The fields are replaced by their value when the e-mail is triggered.



Account: GMail
To: nic.gorleer@iba-ag.com
Cc:
Bcc:
Subject: Test from pda
Text:

```
Last start user:
PC name: [CL:START:PC_NAME]
PC IP address: [CL:START:PC_IP]
Windows user: [CL:START:USER_WIN]
ibaPDA user: [CL:START:USER_PDA]
Reason: [CL:START:REASON]
Time: [CL:START:DATETIME]

Last stop user:
PC name: [CL:STOP:PC_NAME]
PC IP address: [CL:STOP:PC_IP]
Windows user: [CL:STOP:USER_WIN]
ibaPDA user: [CL:STOP:USER_PDA]
Reason: [CL:STOP:REASON]
Time: [CL:STOP:DATETIME]
```

Attachment: Add attachments by dragging image triggers or text signals and dropping them here.
E-mail trigger: 3.0: start trigger Dead time: 0 s
Send retry period: 00:05:00
Total retry time: 01:00:00
☑ Mark retries with [Delayed message ...] in subject

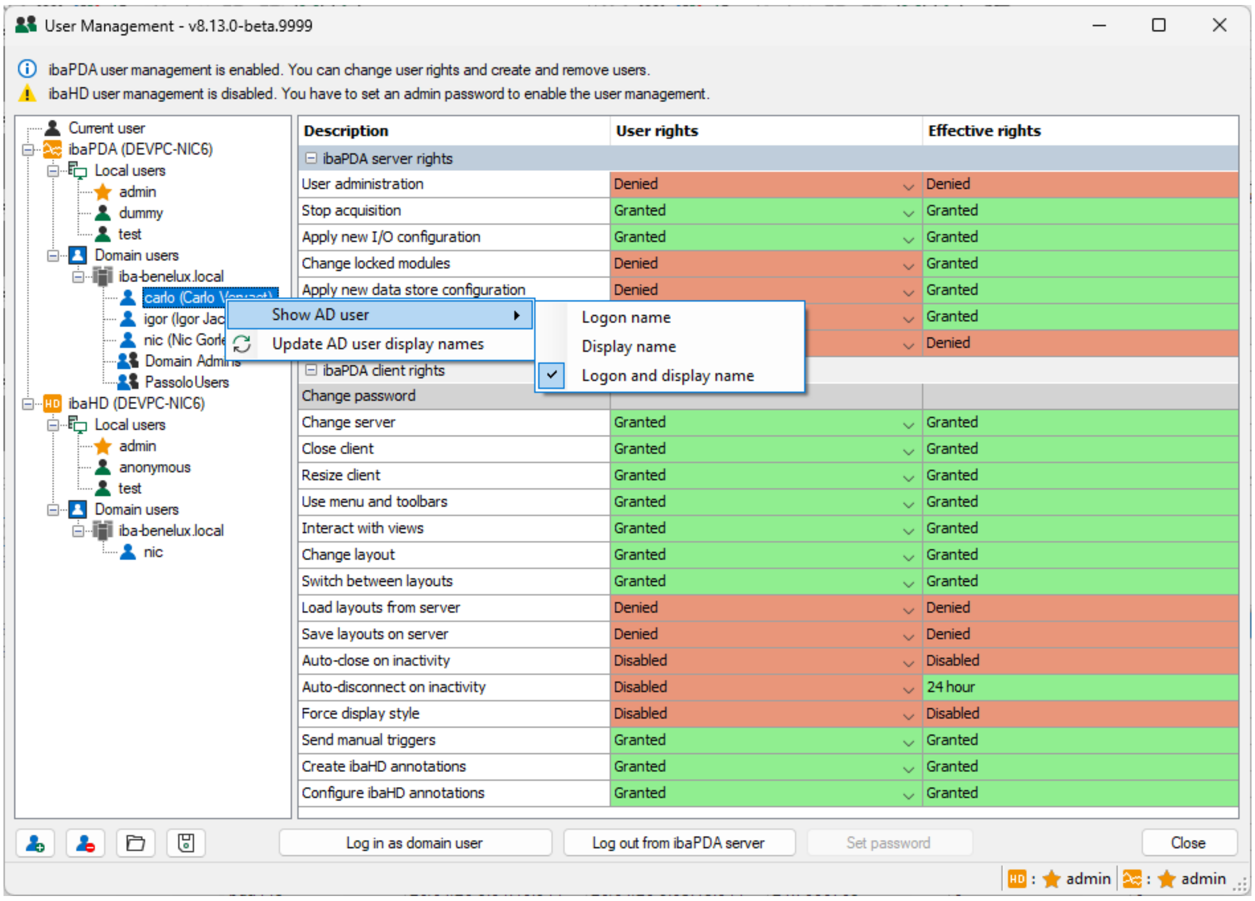
In ibaPDA 8.13.0 new fields have been added. There are 5 fields for the client which last started the acquisition and 5 fields for the client which last stopped the acquisition. The 5 fields are:

- PC name: The name of the client PC.
- PC IP address: The IP address of the client PC.
- Windows user: The Windows user name that was used to log into Windows on the client.
- ibaPDA user: The ibaPDA user name that was used to log into ibaPDA.
- Reason: The reason why the acquisition started or stopped. Possible reasons are:
    - Interactive start
    - Interactive stop
    - Interactive apply new configuration
    - Automatic start
    - Start after remote configuration
    - Start triggered by other synchronized system
    - ibaPDA service is shutting down
    - PC is going to sleep
    - Error occurred
    - License changed
    - …
- Date and time: The date and time when the acquisition started or stopped.

5 User management improvements

5.1 Support for Active Directory display names

Active Directory users have different names. There are the logon name and the display name. In all previous versions the logon name was used in ibaPDA and ibaHD-server. In ibaPDA v8.13.0 the display name can also be shown in the user management form.


User Management - v8.13.0-beta.9999

ibaPDA user management is enabled. You can change user rights and create and remove users.
ibaHD user management is disabled. You have to set an admin password to enable the user management.

- Current user
- ibaPDA (DEVPC-NIC6)
  - Local users
    - admin
    - dummy
    - test
  - Domain users
    - iba-benelux.local
      - carlo (Carlo Vanrast)
      - igor (Igor Jac
      - nic (Nic Gorle
      - Domain Admins
      - PassoloUsers
- ibaHD (DEVPC-NIC6)
  - Local users
    - admin
    - anonymous
    - test
  - Domain users
    - iba-benelux.local
      - nic

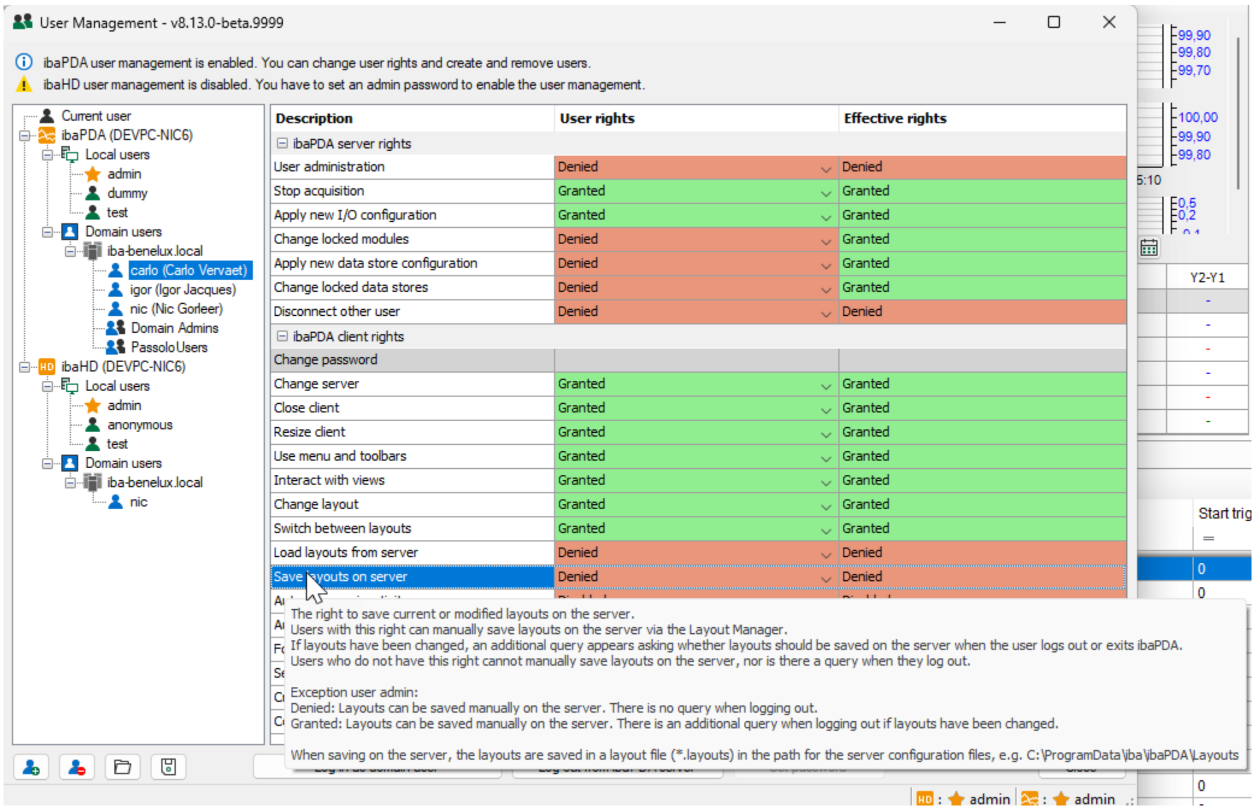
Description	User rights	Effective rights
⊟ ibaPDA server rights		
User administration	Denied	Denied
Stop acquisition	Granted	Granted
Apply new I/O configuration	Granted	Granted
Change locked modules	Denied	Granted
Apply new data store configuration	Denied	Granted
		Granted
		Denied
⊟ ibaPDA client rights		
Change password		
Change server	Granted	Granted
Close client	Granted	Granted
Resize client	Granted	Granted
Use menu and toolbars	Granted	Granted
Interact with views	Granted	Granted
Change layout	Granted	Granted
Switch between layouts	Granted	Granted
Load layouts from server	Denied	Denied
Save layouts on server	Denied	Denied
Auto-close on inactivity	Disabled	Disabled
Auto-disconnect on inactivity	Disabled	24 hour
Force display style	Disabled	Disabled
Send manual triggers	Granted	Granted
Create ibaHD annotations	Granted	Granted
Configure ibaHD annotations	Granted	Granted

IbaPDA will retrieve the display name of new Active Directory users that are added. For existing Active Directory users, you can let ibaPDA retrieve the display name by executing the command "Update AD user display names".

Via the context menu you can also decide which name is shown in the tree:

- Logon name: The logon name is shown, and the display name is shown as tooltip
- Display name: The display name is shown, and the logon name is shown as tooltip
- Logon and display name: The logon name is shown followed by the display name between brackets.

5.2 Improved help texts for user rights



User Management - v8.13.0-beta.9999

ibaPDA user management is enabled. You can change user rights and create and remove users.
ibaHD user management is disabled. You have to set an admin password to enable the user management.

Description	User rights	Effective rights
ibaPDA server rights		
User administration	Denied	Denied
Stop acquisition	Granted	Granted
Apply new I/O configuration	Granted	Granted
Change locked modules	Denied	Granted
Apply new data store configuration	Denied	Granted
Change locked data stores	Denied	Granted
Disconnect other user	Denied	Denied
ibaPDA client rights		
Change password		
Change server	Granted	Granted
Close client	Granted	Granted
Resize client	Granted	Granted
Use menu and toolbars	Granted	Granted
Interact with views	Granted	Granted
Change layout	Granted	Granted
Switch between layouts	Granted	Granted
Load layouts from server	Denied	Denied
Save layouts on server	Denied	Denied

The right to save current or modified layouts on the server.
Users with this right can manually save layouts on the server via the Layout Manager.
If layouts have been changed, an additional query appears asking whether layouts should be saved on the server when the user logs out or exits ibaPDA.
Users who do not have this right cannot manually save layouts on the server, nor is there a query when they log out.

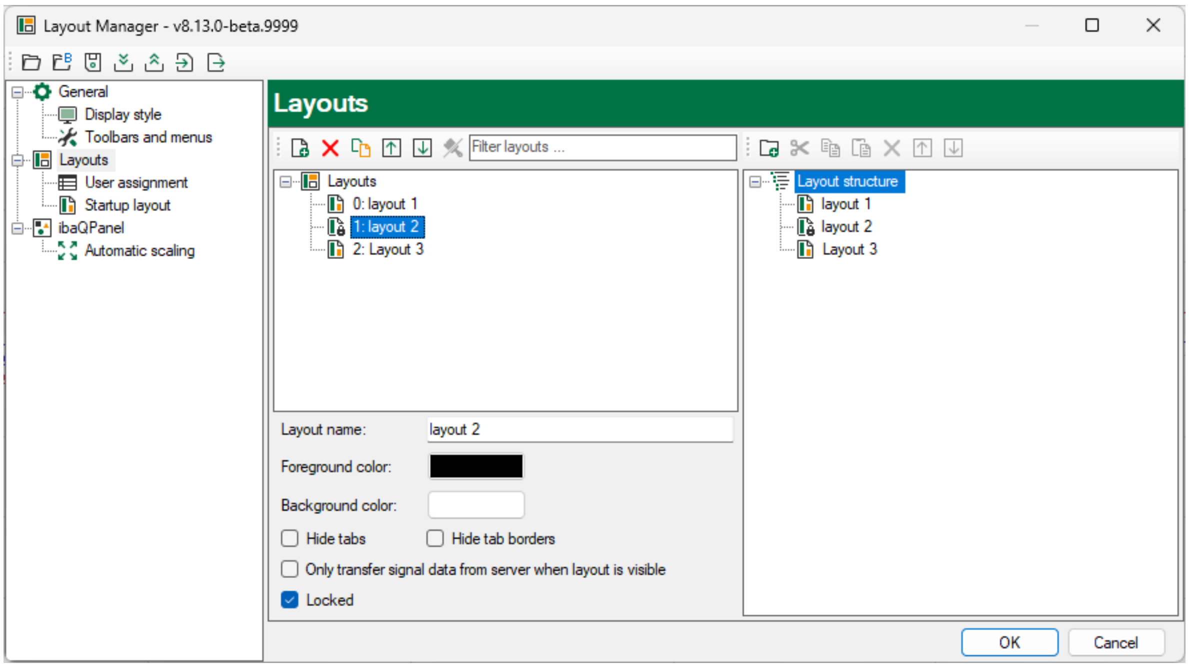
Exception user admin:
Denied: Layouts can be saved manually on the server. There is no query when logging out.
Granted: Layouts can be saved manually on the server. There is an additional query when logging out if layouts have been changed.

When saving on the server, the layouts are saved in a layout file (*.layouts) in the path for the server configuration files, e.g. C:\ProgramData\iba\ibaPDA\Layouts

When you hover over a right then a tooltip will appear with the same help text as in the manual and online help.

6 Locking layouts

There are user rights to prevent a user from changing anything to the layout. This right applies to all layouts. There can be scenarios where you have some layouts that shouldn't be changed and other layouts that the user should change. In order to support such a scenario, the concept of a locked layout has been introduced. A locked layout is a layout that no user can change. If you want to change it then you first have to unlock it.



Layout name: layout 2

Foreground color:

Background color:

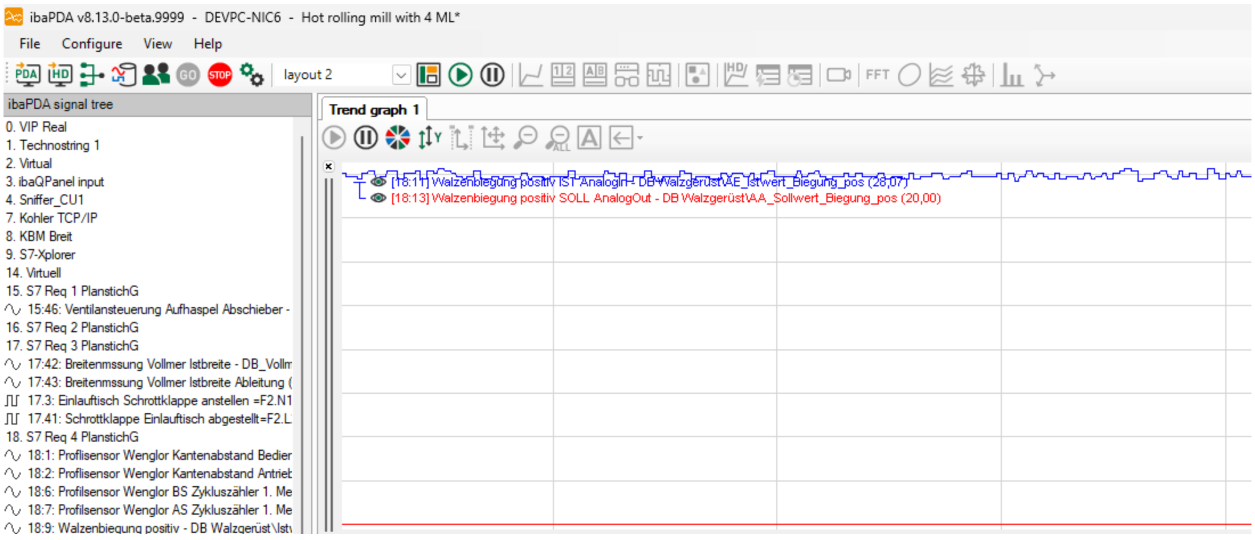
☐ Hide tabs ☐ Hide tab borders

☐ Only transfer signal data from server when layout is visible

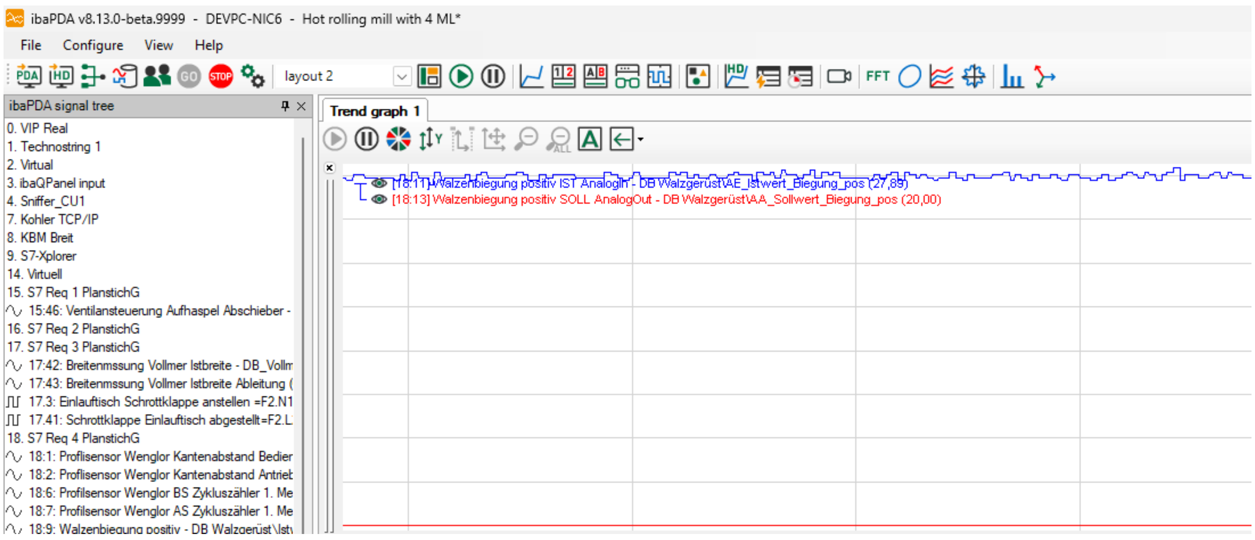
☑ Locked

In the layout manager you can lock or unlock a layout using the "Locked" checkbox. Locked layouts have a lock sign on their icon.

The following screenshot shows a locked layout.


ibaPDA v8.13.0-beta.9999 - DEVPC-NIC6 - Hot rolling mill with 4 ML*
File
Configure
View
Help
layout 2
ibaPDA signal tree
0. VIP Real
1. Technostring 1
2. Virtual
3. ibaQPanel input
4. Sniffer_CU1
7. Kohler TCP/IP
8. KBM Breit
9. S7-Xplorer
14. Virtuell
15. S7 Req 1 PlanstichG
15:46: Ventilansteuerung Aufhaspel Abschieber -
16. S7 Req 2 PlanstichG
17. S7 Req 3 PlanstichG
17:42: Breitenmssung Vollmer Istbreite - DB_Vollm
17:43: Breitenmssung Vollmer Istbreite Ableitung (
17.3: Einlauftisch Schrottklappe anstellen =F2.N1
17.41: Schrottklappe Einlauftisch abgestellt=F2.L
18. S7 Req 4 PlanstichG
18:1: Profilsensor Wenglor Kantenabstand Bedier
18:2: Profilsensor Wenglor Kantenabstand Antriet
18:6: Profilsensor Wenglor BS Zykluszähler 1. Me
18:7: Profilsensor Wenglor AS Zykluszähler 1. Me
18:9: Walzenbiegung positiv - DB Walzgerüst\Ist
Trend graph 1
[18:11] Walzenbiegung positiv IST Analogin - DB Walzgerüst\AE_Istwert_Biegung_pos (28,07)
[18:13] Walzenbiegung positiv SOLL AnalogOut - DB Walzgerüst\AA_Sollwert_Biegung_pos (20,00)

The next screenshot shows the unlocked state.


ibaPDA v8.13.0-beta.9999 - DEVPC-NIC6 - Hot rolling mill with 4 ML*
File
Configure
View
Help
layout 2
FFT
ibaPDA signal tree
Trend graph 1

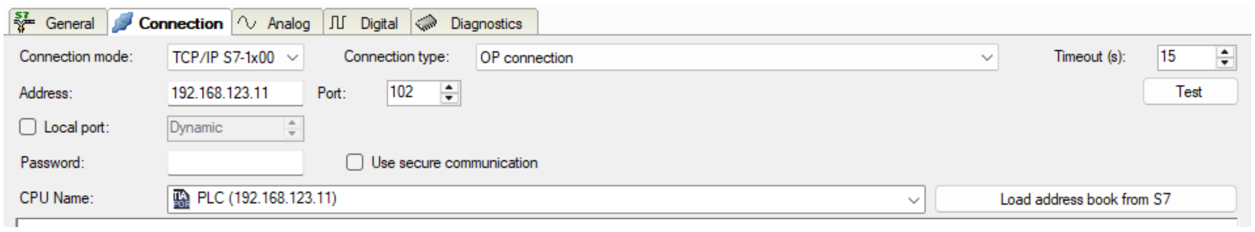
0. VIP Real
1. Technostring 1
2. Virtual
3. ibaQPanel input
4. Sniffer_CU1
7. Kohler TCP/IP
8. KBM Breit
9. S7-Xplorer
14. Virtuell
15. S7 Req 1 PlanstichG
15:46: Ventilansteuerung Aufhaspel Abschieber -
16. S7 Req 2 PlanstichG
17. S7 Req 3 PlanstichG
17:42: Breitenmssung Vollmer Istbreite - DB_Vollm
17:43: Breitenmssung Vollmer Istbreite Ableitung (
17.3: Einlauftisch Schrottklappe anstellen =F2.N1
17.41: Schrottklappe Einlauftisch abgestellt=F2.L
18. S7 Req 4 PlanstichG
18:1: Proflisensor Wenglor Kantenabstand Bedier
18:2: Proflisensor Wenglor Kantenabstand Antriet
18:6: Profilsensor Wenglor BS Zykluszähler 1. Me
18:7: Profilsensor Wenglor AS Zykluszähler 1. Me
18:9: Walzenbiegung positiv - DB Walzgerüst\Istv

[18:11] Walzenbiegung positiv IST AnalogIn - DB Walzgerüst\AE_Istwert_Biegung_pos (27,89)
[18:13] Walzenbiegung positiv SOLL AnalogOut - DB Walzgerüst\AA_Sollwert_Biegung_pos (20,00)

7 TCP port used by S7 connections

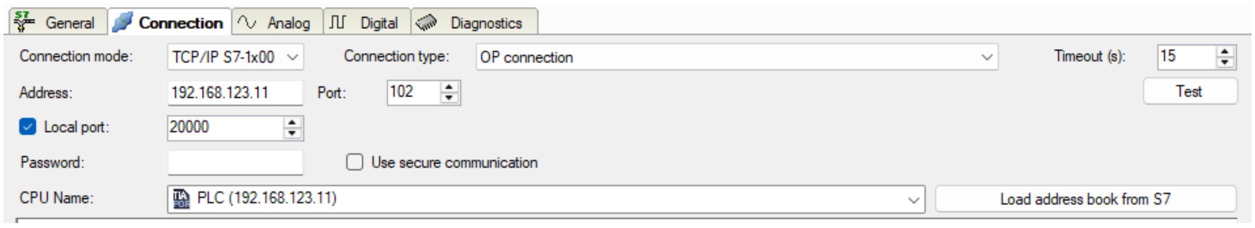
The local TCP port used to connect to an S7 is normally dynamically determined by Windows. For some advanced firewalls this is a problem. They require a fixed source port for the connection. This can now be configured.

Dynamic:



Connection mode: TCP/IP S7-1x00
Connection type: OP connection
Timeout (s): 15
Address: 192.168.123.11
Port: 102
☐ Local port: Dynamic
Password:
☐ Use secure communication
CPU Name: PLC (192.168.123.11)

Fixed:



Connection mode: TCP/IP S7-1x00
Connection type: OP connection
Timeout (s): 15
Address: 192.168.123.11
Port: 102
☑ Local port: 20000
Password:
☐ Use secure communication
CPU Name: PLC (192.168.123.11)