Infor Forcam MES Physical Terminal Identifier

Version 5.11

Publication Information

Release: Infor Forcam MES 5.11

Publication date: September 21, 2022

Contents

About this guide

Intended audience

Organization

This table shows the chapters of the guide:

Section	Description

Related documents

You can find the documents in the product documentation section of the Infor Support Portal, as described in "Contacting Infor" on page 5.

Contacting Infor

If you have questions about Infor products, go to Infor Concierge at https://concierge.infor.com/ and create a support incident.

The latest documentation is available from docs.infor.com or from the Infor Support Portal. To access documentation on the Infor Support Portal, select **Search > Browse Documentation**. We recommend that you check this portal periodically for updated documentation.

If you have comments about Infor documentation, contact documentation@infor.com.

Chapter 1 Concept

With this identifier it is possible to identify and address a specific terminal template running on a dedicated physical machine (PC, laptop etc.).

In order to use this, the following is necessary: a simple text file (UTF-8) has to be created on the relevant client machine . This text file contains an arbitrary string uniquely identifying this physical machine in some way, e.g. "machine_4711". This text file has to be stored in the "client side local directory" of the client machine.

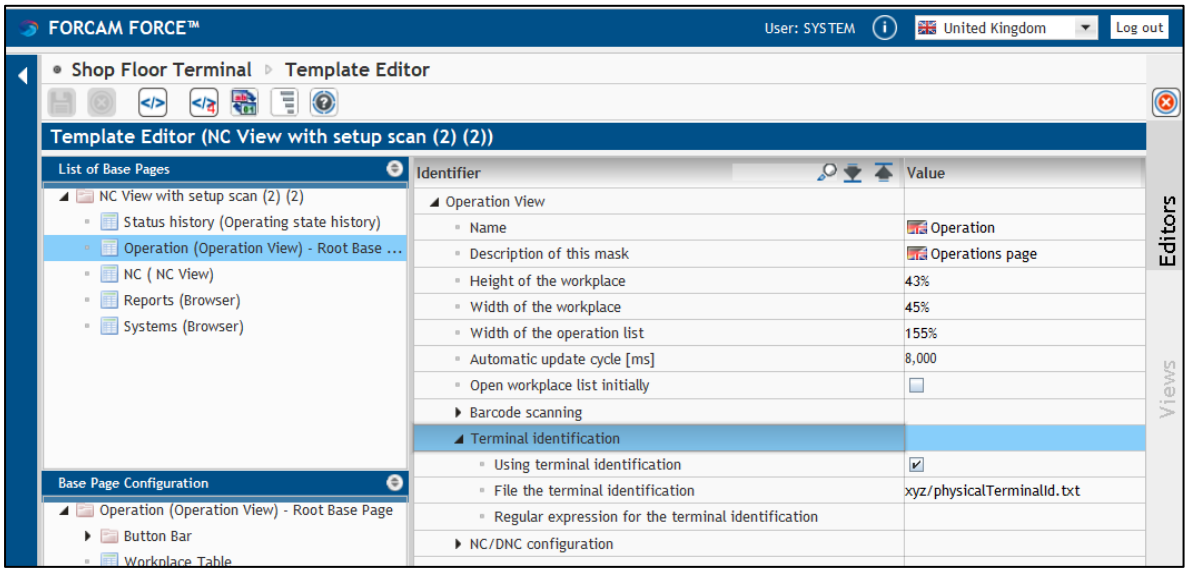
Chapter 2 Configuration

The usage of the identifier is done within the SFT *OperationView* base page with the following three attributes:

1 Use physical terminal identifier - Enable/disable this feature.

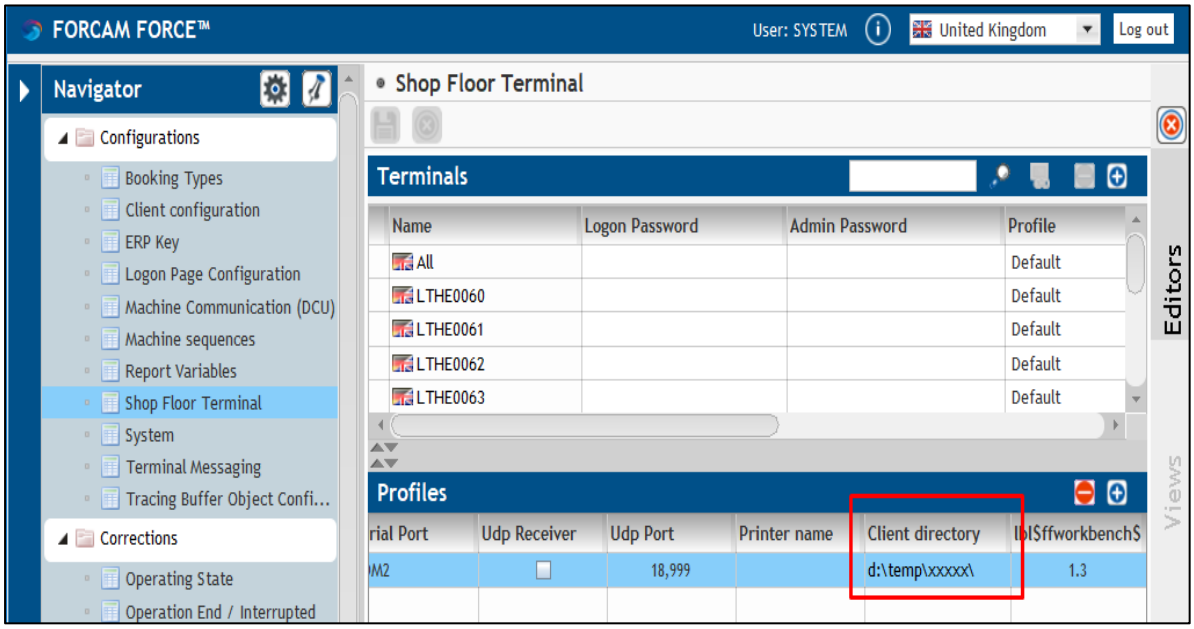
2 File name of the text file on client machine containing the identifier. Possible is a file name only like "physicalTerminalId.txt" or with a relative directory path given, e.g. "xyz/physicalTerminalId.txt". Provide always without initial slash.
This file is relative within the given "client side local directory" configured in terminal profile (see below).

3 Optional: a RegEx to be used parsing the relevant sub string from the text file.

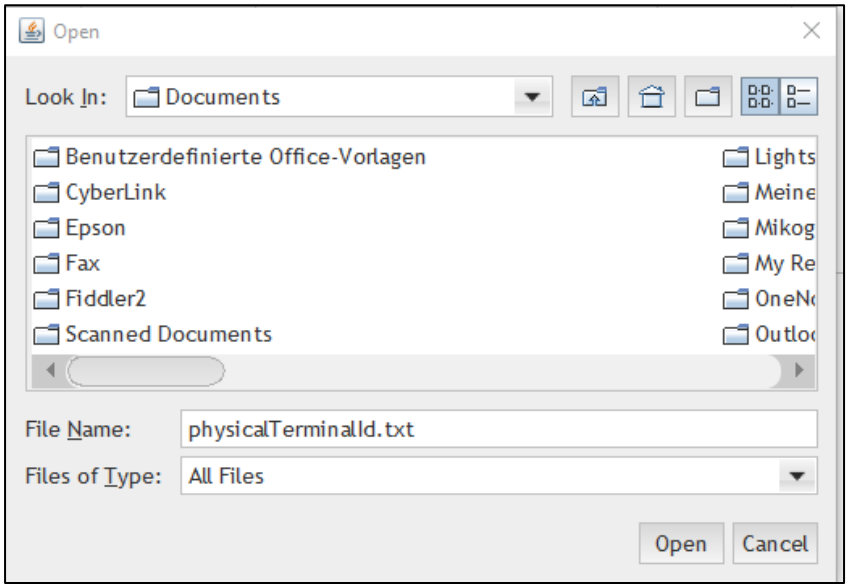


Configuration of the "client side local directory" in terminal profile via SFT-Configurator in the Workbench

The "client side local directory" is a special directory on the client side machine running a template using this terminal profile. Only within this directory the client browser does have access to the local machine (security reasons). The directory can be configured in the terminal profile with attribute "Client directory". An absolute path has to be given, e.g. "d:\temp\xxxxx\".
If no special path configured, the default directory in the users temp directory of OS is used: "C:/Users/UserX/AppData/Local/Temp/eclnt/local/".



Using the values as mentioned above, the resulting path of the terminal identifier file is "d:\temp\xxxxx\xyz\physicalTerminalId.txt". If something is wrong with the path and the file could not be found, a file chooser dlg is coming up shortly after entering the *OperationView;* the file could be chosen manually.



Chapter 3 Address a Terminal on a Specific Physical Machine

Actual usage is to selectively address background activities on terminals running at a specific physical machine. Used e.g. by the DACQ to address a specific client terminal and trigger a background activity.

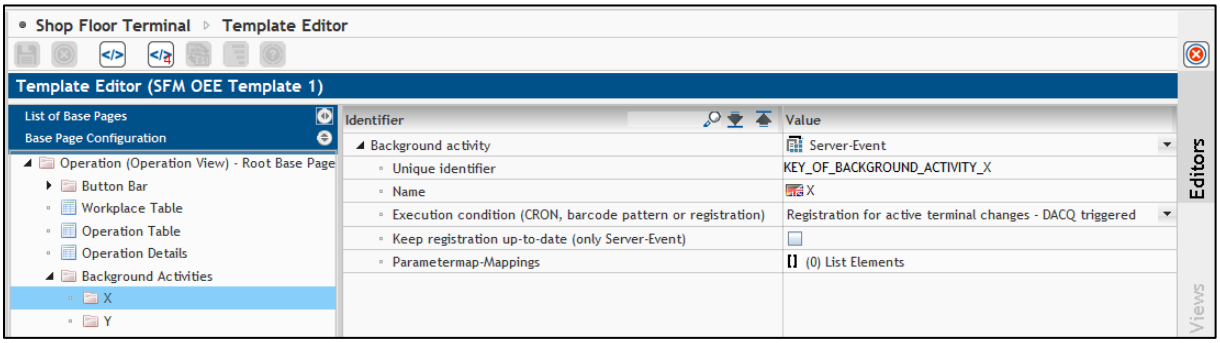
This is done by using the *Server Event* principle, i.e. a specific *server event* has to be send to the SFT. This *server event* contains a unique ID consisting of the physical terminal identifier and the key of the background activity to be executed.
Technically the unique server event ID is a hash value calculated as e.g. `("machine_4711" + "KEY_OF_BACKGROUND_ACTIVITY_X").hashCode()`. Both, the sender (e.g. DACQ) and receiver (SFT) has to know and use this value. Furthermore, the server event scope is relevant. Both create the registration ID together.

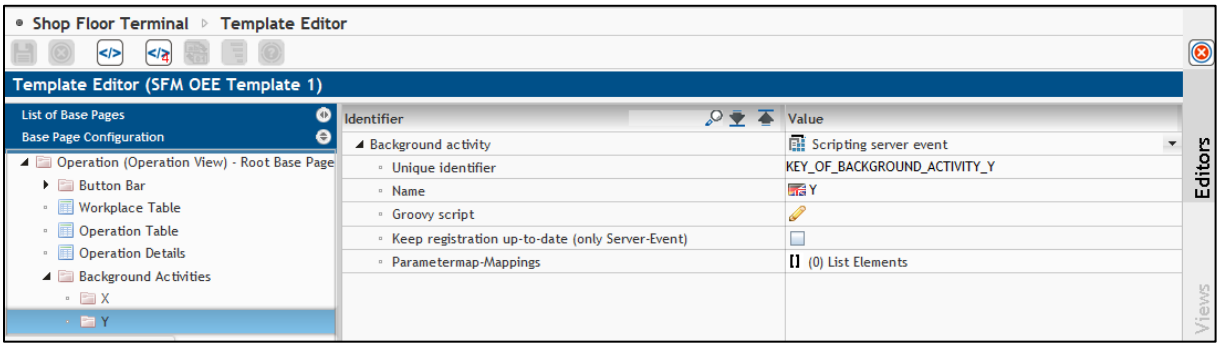
On SFT side, an appropriate background activity of the type "ServerEvent" has to be configured accordingly with the execution condition "Registration for active terminal change - DACQ triggered" (`DACQActivityExecutor`); this registration automatically takes the physical terminal identifier into accout. The DACQ (in fact anyone providing a server event with scope DACQ) as consequence then can trigger this background activity by sending a server event with the corresponding registration ID (see above).

Also a more generic registration is available: the background activity of the type "Scripting server event" (`ScriptingServerEventActivitySetupExecutor`). Via groovy script, an arbitrary registration ID can be calculated; of course also the physical terminal identifier can be taken into account, but also using additionally more criterions like wp etc. and different server event scopes etc.

In contexts using the physical terminal identifier, possibly also the background activity of the type "Scripting based CRON" could be useful. Via a CRON expression it can be calculated via scripting arbitrary whether the corresponding background activity must be executed or not - taking the physical identifier into account.

The following screenshots are configuration examples.


Shop Floor Terminal
Template Editor
Template Editor (SFM OEE Template 1)
List of Base Pages
Base Page Configuration
Operation (Operation View) - Root Base Page
Button Bar
Workplace Table
Operation Table
Operation Details
Background Activities
X
Y
Identifier
Value
Background activity
Server-Event
Unique identifier
KEY_OF_BACKGROUND_ACTIVITY_X
Name
X
Execution condition (CRON, barcode pattern or registration)
Registration for active terminal changes - DACQ triggered
Keep registration up-to-date (only Server-Event)
Parametermap-Mappings
(0) List Elements
Editors
Views


Shop Floor Terminal
Template Editor
Template Editor (SFM OEE Template 1)
List of Base Pages
Base Page Configuration
Operation (Operation View) - Root Base Page
Button Bar
Workplace Table
Operation Table
Operation Details
Background Activities
X
Y
Identifier
Value
Background activity
Scripting server event
Unique identifier
KEY_OF_BACKGROUND_ACTIVITY_Y
Name
Y
Groovy script
Keep registration up-to-date (only Server-Event)
Parametermap-Mappings
(0) List Elements
Editors
Views