FORCAM™ FORCE

Version 5.10

Label Druck

Handbuch

Dokument:	**Handbuch - Label Druck**
Erstellt:	**09.09.16**
Letzte Änderung:	**30.09.19**
Autor:	**STernes**

Inhaltsverzeichnis

1 Einführung

Die Funktion „Label Druck“ dient im Allgemeinen zur Darstellung von statischen und dynamischen Daten aus FORCAM FORCE™ Software-Modulen und können als elektronisches Dokument erstellt oder mit einem Drucker ausgedruckt werden.

Innerhalb dieses Funktionselements werden dem Systembetreiber und Anwender folgende grundsätzliche Konfigurationsmechanismen bereitgestellt:

- Label Dokumenten-Template
- einbinden von dynamischen Daten in dieses Template
- Konfiguration, bei welchen Ereignissen ein Label-Druck erfolgen soll
- Druckerdefinition

Mit dieser Funktion des Label-Drucks kann sowohl der Lieferant, als auch der zukünftige Systembetreiber dynamische Daten im Produktionsablauf durch Ereignisse direkt im Shop Floor ausdrucken und bereitstellen (beispielsweise für den Logistik- und Materialprozess).

2 Konfigurationselemente Label-Druck

Um einen Label-Druck aus dem Prozessablauf heraus dynamisch anhand von Ereignissen erzeugen zu können, stehen folgende Konfigurationselemente in FORCAM FORCE™ bereit:

- Label-Druck Template:
  - Platzhaltervariablen für dynamischen Inhalt
  - Unterstützung von Barcode
- Konfiguration und Zuordnung von Auslöse-Ereignissen als Trigger:
  - Integration in die System-Template-Definitionen (Shop Floor Terminal-Konfigurator und Terminal-Templates)
- Druckerdefinition

2.1 Label-Druck Template

Um unterschiedlichste Label-Varianten für den Anwender zu unterstützen, kann für jeden Anwendungsfall ein individuelles Label-Template-Dokument erstellt werden.

Übersicht der unterstützen Dokumentenarten in FORCAM FORCE™ zur Definition von Label-Druck-Templates (Label-Druck-Formulare):

Tabelle 1: Übersicht der Dokumentarten als mögliches Label-Druck-Formular

Dokumententyp	Beschreibung
***.RTF**	Das Rich Text Format (RTF) ist ein Dateiformat für Texte und kann als Austauschformat zwischen Textverarbeitungsprogrammen verschiedener Hersteller auf verschiedenen Betriebssystemen dienen. Dateiendung: .rtf MIME-Type: text/rtf, application/rtf[1]

2.1.1 Statische Elemente eines Label-Druck-Templates

Für die Definition eines Label-Formulars können alle statischen Gestaltungsmittel, welche ein *.RTF-Format- und Dokument bieten, verwendet werden.
Exemplarisches Beispiel eines Label-Druck-Formulars mit rein statischen Elementen:

Bild 1: Beispiel eines statischen Label-Druck-Formulars *.RTF

2.1.2 Dynamische Datenelemente eines Label-Druck-Templates

Der häufigste Anwendungsfall eines Labels ist, dass zu einem definierten Ereignis ein statisches Formular mit dynamischen Dateninhalte aus dem gerade laufenden Prozess angereichert und ausgedruckt wird.

Dynamische Dateninhalte können innerhalb eines Label-Templates (*.RTF) beliebig positioniert werden. Diese können in Form von definierten Platzhaltern in dem Template platziert werden.

Tabelle 2: Platzhalterdefintion für dynamische Datenelemente

Name	Beschreibung
@Common_Place_Holder@	**Allgemeine Platzhalterdefinition:** Platzhalter beginnen mit der Zeichendeklaration „@“ und werden mit diesem Zeichen ebenfalls abgeschlossen

Für den Label-Druck sind in FORCAM FORCE™ folgende Elemente als dynamischen Inhalt vorgesehen:

Tabelle 3: Unterstützte Datentypen für dynamische Datenelemente

Type	Beschreibung
Numerisch	Beliebige Zahlenwerte können als dynamischen Inhalt einem Platzhalter zugewiesen und als Zeichen im Label dargestellt werden
Alphanumerisch	Beliebige alphanumerische Werte können als dynamischer Inhalt einem Platzhalter zugewiesen und als Zeichen im Label dargestellt werden
String	Beliebige Buchstaben (strings) können als dynamischer Inhalt einem Platzhalter zugewiesen und als Zeichen im Label dargestellt werden
Barcode	Sowohl rein numerische als auch alphanumerische Textdaten können als dynamischer Inhalt einem Platzhalter zugewiesen und als Barcode im Label dargestellt werden. Hierfür muss im *.RTF der Barcode 128 als FFT-Schriftart verwendet werden und ist auf dem FORCAM FORCE™ Applikationsserver bereitgestellt. Der Barcode 128 ist in der internationalen Norm ISO/IEC 15417 vollständig beschrieben. In FORCAM FORCE™ ist die Ausprägung des Barcodes 128B umgesetzt. (https://www3.hi-tier.de/Entwicklung/Technik/barcode_Code128.html)

Für den Label-Druck stehen in FORCAM FORCE™ folgende Elemente als dynamische Platzhalter mit dem zugehörigen Datenbank-Feld als Quelle zur Verfügung:

Tabelle 4: Übersicht nutzbarer, dynamischer Platzhalter

Platzhalter	Beschreibung
@material$number_number@	Als dynamischer Parameter wird dieser Platzhalter zum Trigger-Zeitpunkt mit der Materialnummer ersetzt (ob diese Daten zur Verfügung stehen, ist bei der Anwendung abhängig von der Operation)
@material$description_description@	Als dynamischer Parameter wird dieser Platzhalter zum Trigger-Zeitpunkt mit der Materialbeschreibung ersetzt (ob diese Daten zur Verfügung stehen, ist bei der Anwendung abhängig von der Operation)
@material$type_type@	Als dynamischer Parameter wird dieser Platzhalter zum Trigger-Zeitpunkt mit dem Materialtyp ersetzt (ob diese Daten zur Verfügung stehen, ist bei der Anwendung abhängig von der Operation)
@erpYieldQuantity@	Als dynamischer Parameter wird dieser Platzhalter zum Trigger-Zeitpunkt mit der Gutmenge ersetzt (ob diese Daten zur Verfügung stehen, ist bei der Anwendung abhängig von der Operation. Die Gutmenge wird dem Ereignis durch die Vorgängeraktivität „Mengenbuchung“ bereitgestellt)
@erpScrapQuantity@	Als dynamischer Parameter wird dieser Platzhalter zum Trigger-Zeitpunkt mit der Ausschussmenge ersetzt (ob diese Daten zur Verfügung stehen, ist bei der Anwendung abhängig von der Operation. Die Ausschussmenge wird dem Ereignis durch die Vorgängeraktivität „Mengenbuchung“ bereitgestellt)
@erpReworkQuantity@	Als dynamischer Parameter wird dieser Platzhalter zum Trigger-Zeitpunkt mit der Nacharbeitsmenge ersetzt (ob diese Daten zur Verfügung stehen, ist bei der Anwendung abhängig von der Operation. Die Nacharbeitsmenge wird dem Ereignis durch die Vorgängeraktivität „Mengenbuchung“ bereitgestellt)

Mithilfe von Prozessinstruktionen können Platzhalter um Funktionen in Templates erweitert werden. Grundsätzlich können mehrere Prozessinstruktionen für einen Platzhalter definiert und aneinandergereiht werden. Aktuell mögliche Prozessinstruktionen sind in der folgenden Übersicht dargestellt:

Tabelle 5: Übersicht der möglichen Prozessinstruktionen

Platzhalter	Beschreibung
|prefix='' **Beispiel:** **@material$number_number|prefix=''@**	Mithilfe dieser Prozessinstruktion kann ein dynamischer Wert immer um ein statisches Präfix **prefix=''** ergänzt werden. Eine Prozessinstruktion beginnt immer mit dem Zeichen **|** und der Prozessinstruktion, gefolgt von dem jeweiligen Prozesswert (hier **prefix='Prozesswert'**). Beendet wird eine Prozessinstruktion entweder durch den Beginn einer weiteren Instruktion mit **|** oder dem Ende der Platzhalterdefinition **@**. Der Prozesswert der Prozessinstruktion muss innerhalb der Zeichen **''** definiert werden. Beispiel eines Character_Präfixtest: @material$number_number**|prefix='Präfixtest'**@ Beispiel einer dynamischen Materialnummer: 123456789

	Ergebnis: Präfixtest123456789
|formatter=formattername **Beispiel:** **@material$number_number|formatter=barcode128B@**	Mithilfe dieser Prozessinstruktion kann ein dynamischer Wert unterschiedlich vorverarbeitet werden. Eine Prozessinstruktion beginnt immer mit dem Zeichen **|** und der Prozessinstruktion, gefolgt von dem jeweiligen Prozesswert (hier **formatter="formattername"**). Beendet wird eine Prozessinstruktion entweder durch den Beginn einer weiteren Instruktion mit **|** oder dem Ende der Platzhalterdefinition **@**. Der Prozesswert dieser Prozessinstruktion muss an der Stelle **formattername** definiert werden. Beispiel: Die Darstellung des dynamischen Inhaltes als Barcode128 wird auf ein spezifisches Format Barcode128B formatiert. @material$number_number**\|formatter=barcode128B**@ Beispiel mit mehrfacher Prozessinstruktionen: @material$number_number**|prefix='Präfixtest'|formatter=barcode128B**@

2.1.3 Exemplarisches Beispiel eines Label-Druck-Templates

Exemplarisches Beispiel für ein Label bei einer Mengenmeldung auf einem aktuell laufenden Auftrag. Auf dem Label sollen folgende Informationen dargestellt werden (Platzhalter rot markiert):

- numerische/alphanumerische Materialnummer
- numerische/alphanumerische Materialnummer als Barcode
- gemeldete Gutmenge

Die folgende Abbildung zeigt das zugehörige *.RTF-Dokument:

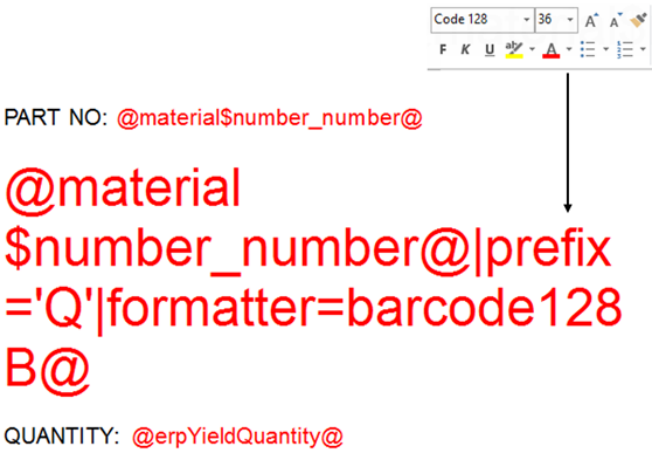

Bild 2: Einfaches, exemplarisches *.RTF-Template mit Platzhalter für dynamischen Inhalt (Rot markiert) und einem Barcode 128-Element als FFT-Schriftart, formatiert als „Barcode 128B")

Bild 3: Ergebnis des Label-Drucks mit gefülltem, dynamischen Inhalt zum Zeitpunkt des Trigger-Ereignisses

2.2 Konfiguration eines Aktivitätsschritts für Label-Druck

Um einen Label-Druck erzeugen zu können, muss für das erstellte Label-Druck-Template (*.RTF) ein Aktivitätsschritt definiert werden.
In FORCAM FORCE™ kann der Label-Druck nur durch Aktionen an einem Shop Floor Terminal erzeugt werden. Die Konfiguration des Shop Floor Terminals erfolgt in der Systemadministration von FORCAM FORCE™ mithilfe eines Shop Floor Terminal Konfigurators und erzeugt die notwendigen Terminal-Templates.

Mit diesem Konfigurator kann ein Label-Druck als ein Aktivitätsschritt definiert werden. Im folgenden Abschnitt ist dies anhand eines realen Beispiels veranschaulicht.

2.2.1 Shop Floor Terminal Konfigurator

Aufruf	**Workbench: Shop Floor Terminal - Konfiguration → Vorlagen - Editor**

Für die ausführliche Beschreibung der Funktionen des Shop Floor Terminal Konfigurators für den Anwender wird auf das Handbuch „Stammdaten und Systemkonfiguration“ von FORCAM FORCE™ verwiesen.

Beispiel
Als exemplarisches Beispiel wird in diesem Dokument das Hinzufügen eines Label-Drucks an die Aktion „Mengenmeldung“ in einer bestehenden Terminal Template-Konfiguration dargestellt.

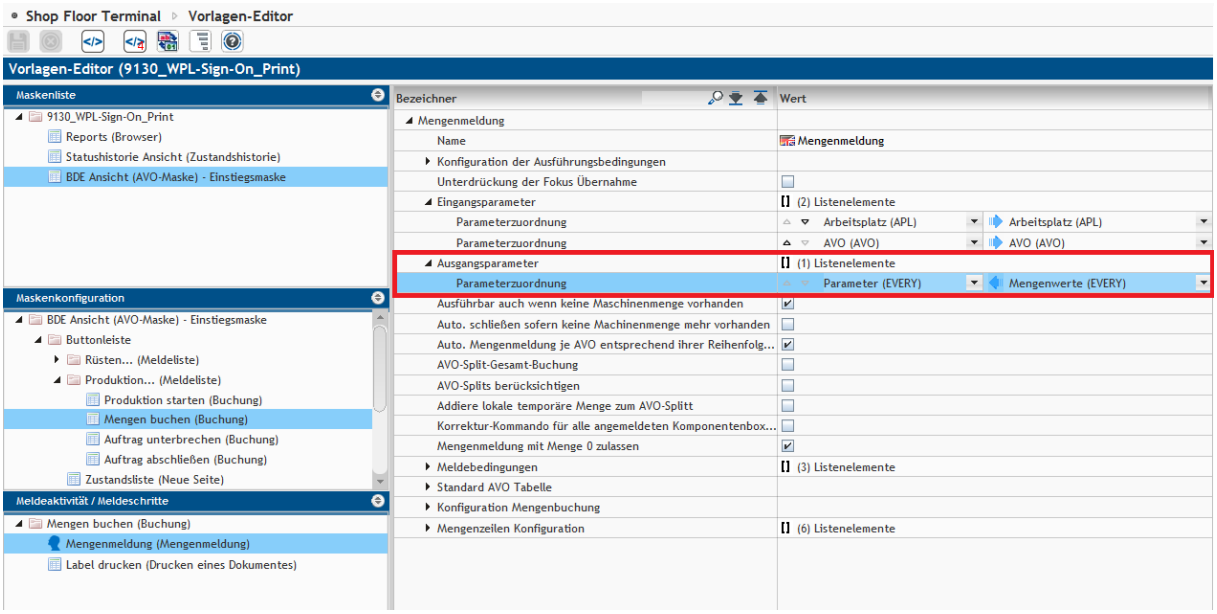

Bild 4: Mengenbuchung Aktivität/Schritt im Shop Floor Terminal Konfigurator

In diesem Label- Druck-Beispiel soll bei einer Mengenbuchung auf einen Auftrag ein Label erzeugt werden. Als Daten für das Label werden sowohl Material als auch Mengendaten benötigt.

Als logische Ausführungsschritte für eine Shop Floor Terminal-Konfiguration ergibt sich damit folgende Prozessstruktur:

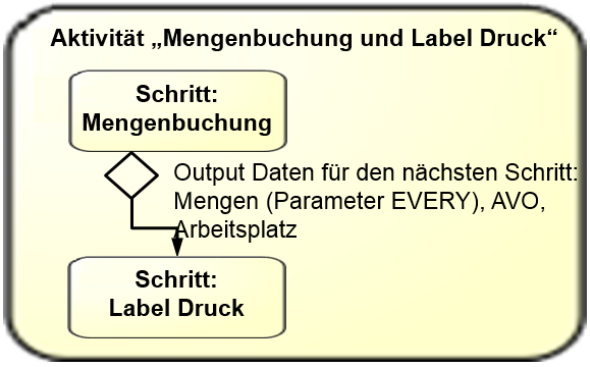

Bild 5: Logische Ausführungsschritte für die Shop Floor Terminal-Konfiguration eines Template für „Mengenbuchung mit Label-Druck“

Als erster Ausführungsschritt erfolgt die Mengenbuchung über einen entsprechenden Mengendialog am Shop Floor Terminal. In diesem Dialog gibt der Werker die entsprechenden Buchungsdaten für dedizierte Aufträge ein. Diese Daten dienen als Output-Daten des Schritts „Mengenbuchung“ und als Input-Daten für einen definierten, nachfolgenden Schritt.

Soll für eine Mengenbuchung ebenfalls ein Aktivitätsschritt Label-Druck erfolgen, wird einfach ein weiterer Schritt „Label-Druck“ angehängt.

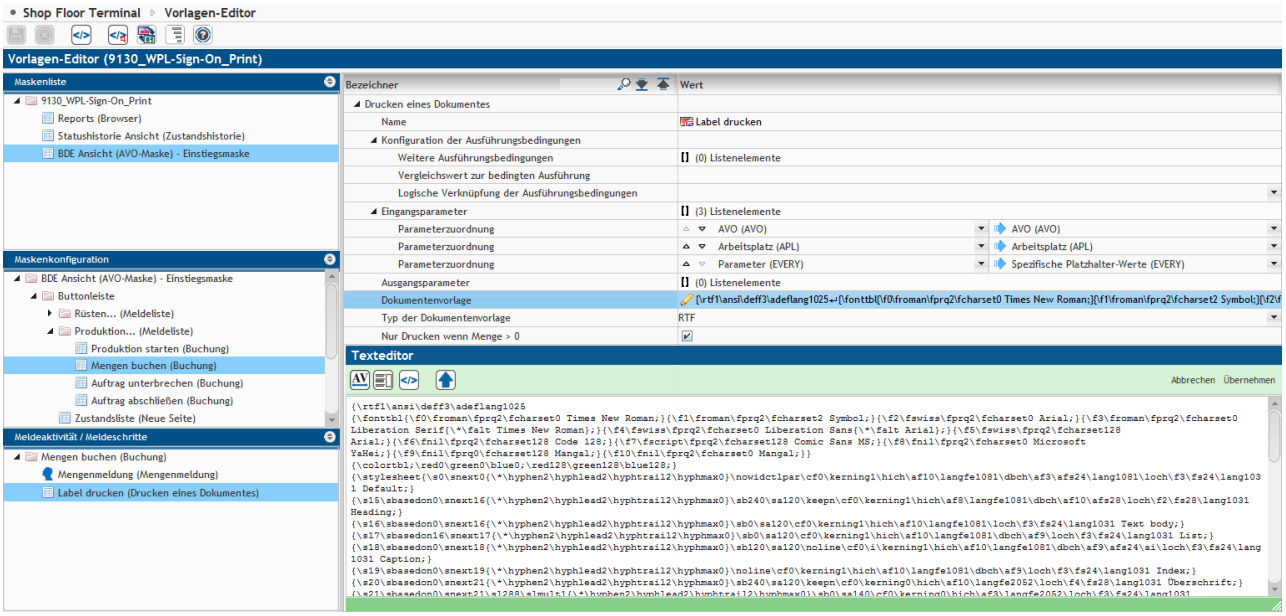

Bild 6: Schritt Label-Druck (Drucken eines Dokuments) im Shop Floor Terminal Konfigurator

Für den Schritt „Label-Druck (Drucken eines Dokumentes)“ können folgende Elemente konfiguriert werden:

Tabelle 6: Übersicht Konfigurationselemente für den Schritt Label-Druck im Shop Floor Terminal Konfigurator

Konfigurationselemente	Beschreibung
Name	Name des Schrittes
Eingangsparameter	Das wichtigste Element. Hier wird definiert, welche Daten aus einem vorherigen Schritt als Eingangsdaten für den aktuellen Schritt zur Verarbeitung bereitgestellt werden. In dem hier beschriebenen Beispiel: – AVO – Arbeitsplatz Parameter EVERY (spezifische Platzhalter-Werte) →Definition Ausgangswerte des Schrittes „Mengenbuchung” (Mengenwerte (EVERY) vgl. Bild 4)
Ausgangsparameter	Ebenfalls ein wichtiges Element. Hier wird definiert, welche Daten aus dem aktuellen Schritt als Ausgangsdaten bereitgestellt werden. Ausgangsdaten können für einen nachfolgenden Schritt als Input-Daten bereitgestellt werden. Hinterlegen/Upload des Label-Druck-Templates (Dokument), welches bei der Ausführung des Schritts ausgedruckt werden soll.
Dokumentenvorlage	Welcher Arbeitsplatz welche Druck-Templates verwenden soll, kann über unterschiedliche Shop Floor Terminal-Templates abgebildet werden (Zuweisung Arbeitsplatz -> Shop Floor Terminal-Template).

	Bei Anwählen des Attributes wird das Vorlagendokument in einem Texteditor geöffnet (als reine Datendarstellung).
Typ der Dokumentenvorlage	Definition des Dokumenten-Typs des Label-Druck-Templates
Nur Drucken wenn Menge > 0	Aktiviert/Deaktiviert: Nur bei Mengenbuchungen der Gutmenge > 0 wird ein Label-Druck durchgeführt

2.3 Zuweisung von Terminal-Templates und Drucker

Um ein mit dem Shop Floor Terminal Konfigurator erstelltes Terminal-Template anzuwenden, müssen dem Terminal die gewünschten Arbeitsplätze zugeordnet werden.

Aufruf	**Workbench: Konfiguration → Shop Floor Terminal**

Bei der Zuordnung kann jeweils ein expliziter Drucker einem Arbeitsplatz zugeordnet werden, auf dem die Labels ausgedruckt werden sollen.
Als Name des Druckers muss der exakte Windows Druckername eingetragen werden.

Wird kein expliziter Drucker gepflegt, wird automatisch der auf dem Ausführungsrechner als „Default“ deklarierte Drucker für den Druck verwendet. Ist dieser ebenfalls nicht vorhanden, wird der Drucker-Auswahl-Dialog als Popup für eine manuelle Auswahl angezeigt.

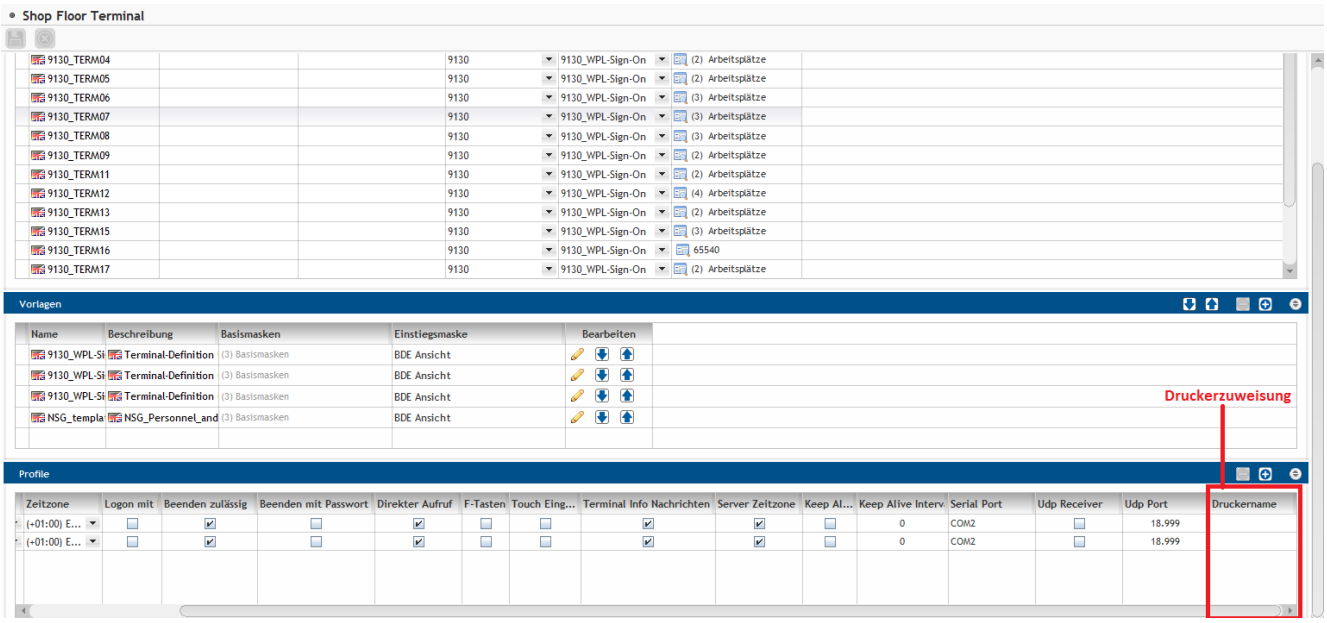

Bild 7: Explizite oder Default-Druckerdefinition bei der Shop Floor Terminal-Zuordnung

3 Barcode Arten und Scan-Methoden

FORCAM FORCE™ unterstützt mehrere Methoden für das Scannen (Lesen) von Barcodes und Drucken von Etiketten oder Dokumenten am Terminal. Diese Methoden sind in der Barcode Bibliothek gesammelt. Die Barcode Bibliothek beinhaltet zudem viele verschiedene Arten von Barcodes, wie z. B. Linear, 2D, QR, PDF417, Data Matrix und so weiter.

3.1 Barcode-Arten

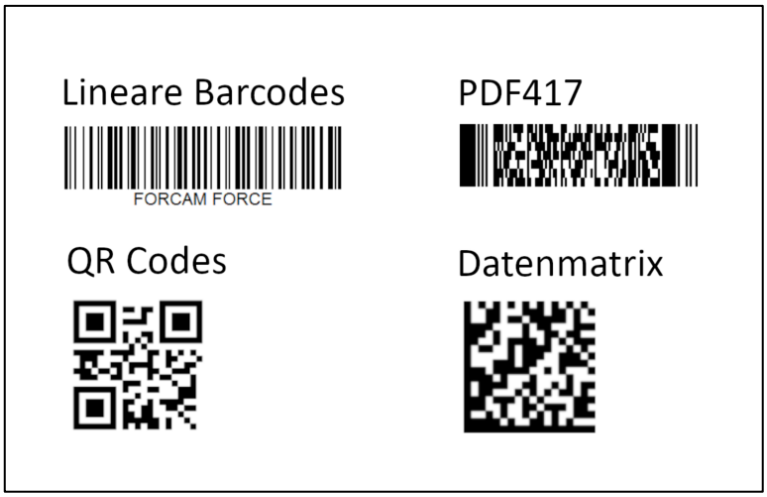

Bild 8: Beispiel von Barcode-Arten

ⓘ Das Verwenden von Fonts für Barcodes wurde durch die Barcode Bibliothek abgelöst.

3.2 Barcode Scan-Methoden

Es gibt verschiedene Barcode Scan-Methoden, die je nach Bedarf konfiguriert werden können.
Die Konfiguration wird in der Workbench unter der Rubrik **Shop Floor Terminal** - **Vorlagen-Editor – Maskenkonfiguration – BDE-Ansicht (AVO Maske)** vorgenommen. Weitere Informationen zur Konfiguration der Barcode Scan-Methode sind im Handbuch Stammdaten und Systemkonfiguration beschrieben.

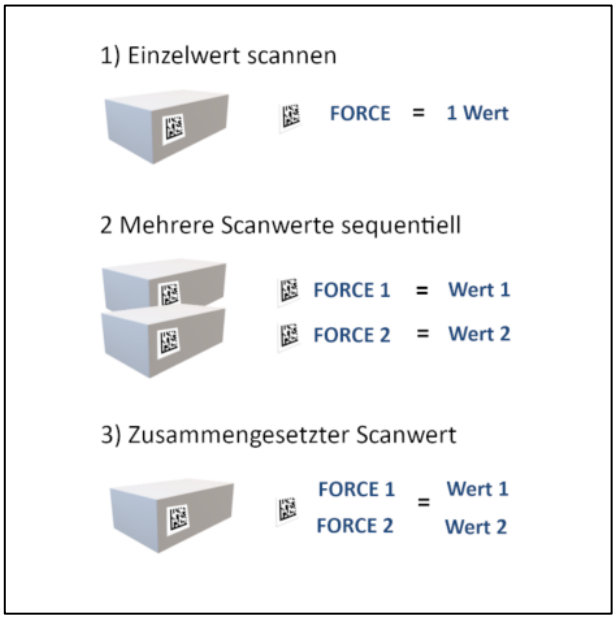

Bild 9: Barcode Scan-Methoden

1) Einzelwert scannen	Liest ein Barcode und gibt einen Wert per Scan aus. Das heißt, ein Barcode gibt einen Wert zurück und kann, je nach Konfiguration, eine Hintergrundaktivität auslösen.
2) Mehrere Scanwerte sequentiell	Es werden mehrere Scanwerte sequentiell via Dialog gescannt. Das heißt, mehrere Barcode-Scans werden mit jeweils einem Wert direkt hintereinander eingescannt. Dazu wird automatisch ein Wert nach dem Anderen abgefangen. Die Sequenz wird im Shop Floor Terminal definiert.
3) Zusammengesetzter Scanwert	Barcodes mit jeweils mehreren Werten können aufgeteilt und ihre Werte auf verschiedene Variablen abgebildet werden. Dies ist zum Beispiel sinnvoll, wenn verschiedene Materialien auf einen Arbeitsplatz montiert werden. Da ein Barcode selbst nicht die Information enthält, ob es sich um einen Hauptteil oder eine Komponente handelt, kann es hier definiert werden.

4 Anhang

Das *.RTF Datenformat stellt den Inhalt wie folgt dar (Beispiel Bild 2):

{\rtf1\ansi\deff3\adeflang1025
{\fonttbl{\f0\froman\fprq2\fcharset0 Times New Roman;}{\f1\froman\fprq2\fcharset2 Symbol;}{\f2\fswiss\fprq2\fcharset0 Arial;}{\f3\froman\fprq2\fcharset0 Liberation Serif{*\falt Times New Roman};}{\f4\fswiss\fprq2\fcharset0 Liberation Sans{*\falt Ari-al};}{\f5\fswiss\fprq2\fcharset1 Arial;}{\f6\fnil\fprq2\fcharset1 Code 128;}{\f7\fnil\fprq2\fcharset0 Microsoft YaHei;}{\f8\fnil\fprq2\fcharset0 Man-gal;}{\f9\fnil\fprq0\fcharset1 Mangal;}}
{\colortbl;\red0\green0\blue0;\red128\green128\blue128;}
{\stylesheet{\s0\snext0\nowidctlpar{*\hyphen2\hyphlead2\hyphtrail2\hyphmax0}\cf0\kerning1\dbch\af10\langfe2052\dbch\af8\afs24\alang1081\loch\f3\fs24\lang1031 Normal;}
{\s15\sbasedon0\snext16\sb240\sa120\keepn\dbch\af7\dbch\af8\afs28\loch\f4\fs28 \u220\'dcberschrift;}
{\s16\sbasedon0\snext16\sl288\slmult1\sb0\sa140 Textk\u246\'f6rper;}
{\s17\sbasedon16\snext17\sl288\slmult1\sb0\sa140\dbch\af9 Liste;}
{\s18\sbasedon0\snext18\sb120\sa120\noline\i\dbch\af9\afs24\ai\fs24 Beschriftung;}
{\s19\sbasedon0\snext19\noline\dbch\af9 Verzeichnis;}
}{\info{\creatim\yr2015\mo8\dy15\hr8\min40}{\revtim\yr0\mo0\dy0\hr0\min0}{\printim\yr0\mo0\dy0\hr0\min0}{\comment LibreOffice}{\vern67241218}}\deftab709
\viewscale90
{*\pgdsctbl
{\pgdsc0\pgdscuse451\pgwsxn11906\pghsxn16838\marglsxn1134\margrsxn1134\margtsxn1134\margbsxn1134\pgdscnxt0 Standard;}}
\formshade\paperh16838\paperw11906\margl1134\margr1134\margt1134\margb1134\sectd\sbknone\sectunlocked1\pgndec\pgwsxn11906\pghsxn16838\marglsxn1134\margrsxn1134\margtsxn1134\margbsxn1134\ftnbj\ftnstart1\ftnrstcont\ftnnar\aenddoc\aftnrstcont\aftnstart1\aftnnrlc
\pgndec\pard\plain \s0\nowidctlpar{*\hyphen2\hyphlead2\hyphtrail2\hyphmax0}\cf0\kerning1\dbch\af10\langfe2052\dbch\af8\afs24\alang1081\loch\f3\fs24\lang1031{\afs32\rtlch \ltrch\loch\fs32\loch\f5
PART NO: **@material$number_number@**}
\par \pard\plain \s0\nowidctlpar{*\hyphen2\hyphlead2\hyphtrail2\hyphmax0}\cf0\kerning1\dbch\af10\langfe2052\dbch\af8\afs24\alang1081\loch\f3\fs24\lang1031\afs32\rtlch \ltrch\loch\fs32\loch\f5

\par \pard\plain \s0\nowidctlpar{*\hyphen2\hyphlead2\hyphtrail2\hyphmax0}\cf0\kerning1\dbch\af10\langfe2052\dbch\af8\afs24\alang1081\loch\f3\fs24\lang1031{\afs72\rtlch \ltrch\loch\fs72\loch\f6
@material$number_number|prefix='Q'|formatter=barcode128B@}
\par \pard\plain \s0\nowidctlpar{*\hyphen2\hyphlead2\hyphtrail2\hyphmax0}\cf0\kerning1\dbch\af10\langfe2052\dbch\af8\afs24\alang1081\loch\f3\fs24\lang1031\afs32\rtlch \ltrch\loch\fs32\loch\f5

\par \pard\plain \s0\nowidctlpar{*\hyphen2\hyphlead2\hyphtrail2\hyphmax0}\cf0\kerning1\dbch\af10\langfe2052\dbch\af8\afs24\alang1081\loch\f3\fs24\lang1031{\afs32\rtlch\ltrch\loch\fs32\loch\f5
QUANTITY: **@erpYieldQuantity@**}
\par \pard\plain \s0\nowidctlpar{*\hyphen2\hyphlead2\hyphtrail2\hyphmax0}\cf0\kerning1\dbch\af10\langfe2052\dbch\af8\afs24\alang1081\loch\f3\fs24\lang1031\afs32\rtlch\ltrch\loch\fs32\loch\f5