



Installation und Konfiguration von COP Dok

Version 3.01



**Inhaltsverzeichnis**

<b>1</b>	<b>Installation .....</b>	<b>5</b>
1.1	Systemvoraussetzungen .....	5
1.2	Hinweis zu den Lizenzen .....	5
1.3	Vorbereitung .....	5
1.3.1	Informationen beschaffen .....	5
1.3.2	Lizenz in Creo Elements/Direct Lizenzserver (MEIs) eintragen .....	5
1.4	Installation durchführen.....	6
1.5	COP Dok Konfiguration .....	8
<b>2</b>	<b>Konfiguration .....</b>	<b>9</b>
2.1	Konfigurationsdateien.....	9
2.2	lpmi.ini .....	9
2.3	Konventionen für lpmi.ini und default.m .....	9
<b>3</b>	<b>Allgemeine und System Konfigurationen in lpmi.ini .....</b>	<b>11</b>
3.1	loadfile[n] .....	11
3.2	log.config.....	11
3.3	progressbar.start .....	11
3.4	system.env.set. <i>Systemvariablen</i> .....	12
3.5	system.tmp.dir .....	12
<b>4</b>	<b>Creo Elements/Direct Drafting und 2D Access Konfigurationen ....</b>	<b>12</b>
4.1	hiddenline.enable.....	12
4.2	me.dir .....	12
4.3	me.customize.file.....	13
4.4	macro.load.post.....	13
4.5	me.start.drafting .....	13
<b>5</b>	<b>Konfigurationen für Pixmap in Zeichnungen .....</b>	<b>14</b>
5.1	pixmap.quality.....	14
5.2	pixmap.color .....	14
<b>6</b>	<b>Creo Elements/Direct 3D Access und Modeling Konfiguration .....</b>	<b>15</b>
6.1	sd.dir .....	15
6.2	sd.application .....	15
6.3	sd.application.args .....	15
6.4	sd.application.kill.process.onstart .....	16
6.5	sd.application.kill.process.all .....	16
6.6	sd.application.restart.always .....	17
6.7	sd.application.restart.if_not_responding .....	17
6.8	sd.application.maxwait.load .....	<b>Fehler! Textmarke nicht definiert.</b>

6.9	sd.delete.all .....	18
<b>7</b>	<b>Druckerkonfigurationen .....</b>	<b>19</b>
7.1	printer[n].name .....	19
7.2	printer[n].system.name .....	19
7.3	printer[n].format[i] .....	19
7.4	printer[n].format[i].system.format .....	20
7.5	printer[n].format[i].width .....	20
7.6	printer[n].format[i].height .....	20
7.7	printer[n].format[i].xoffset .....	20
7.8	printer[n].format[i].yoffset .....	20
7.9	printer[n].format[i].scale .....	21
7.10	printer[n].format[i].orientation .....	21
7.11	printer[n].center .....	21
7.12	printer[n].rotate .....	21
7.13	printer[n].macro.transformation .....	21
7.14	printer[n].macro.transformation.aN .....	22
7.15	printer.default .....	22
7.16	printer.default.aN .....	22
7.17	printer.macro.load.post .....	22
7.18	printer[n].enable .....	23
<b>8</b>	<b>Konverter Konfigurationen .....</b>	<b>24</b>
8.1	converter[n].name .....	24
8.2	converter[n].type .....	24
8.3	converter[n].default .....	25
8.4	converter[n].file.extension .....	25
8.5	converter[n].macro.transformation .....	26
8.6	converter[n].macro.transformation.aN .....	26
8.7	converter[n].macro.load.post .....	26
8.8	converter[n].macro.post .....	26
8.9	Ausgabe .....	27
8.9.1	converter[n].output.file.name .....	27
8.9.2	converter[n].output.file.name.blank.replaceby .....	27
8.9.3	converter[n].output.file.name.blank.replaceby.no.duplicate .....	27
8.9.4	converter[n].output.file.name.blank.remove .....	28
8.9.5	converter[n].sheets.separator .....	28
8.9.6	converter[n].sheets.prefix .....	28
8.9.7	converter[n].sheets.postfix .....	28
8.9.8	converter[n].sheets.number.add.mode .....	29
8.9.9	converter[n].destination.dir.name .....	29

8.9.10	converter[n].output.format .....	29
8.10	converter[n].delegate[m] .....	30
8.11	converter[n].hide[m].part .....	30
8.12	converter[n].hide[m].info .....	30
8.13	converter[n].pdf.merge .....	31
8.14	converter[n].scale .....	31
8.15	converter[n].hiddenline.enable .....	31
8.16	converter[n].post.cmd .....	32
<b>9</b>	<b>Grafikdateien Ausgabe .....</b>	<b>33</b>
9.1	converter.gs.dir .....	33
9.2	converter[n].density .....	33
9.3	converter[n].geometry .....	33
9.4	converter[n].converter.options .....	33
<b>10</b>	<b>PKM Konfigurationen .....</b>	<b>34</b>
10.1	pkm.changenotes.max .....	34
10.2	pkm.lang .....	34
10.3	pkm.reload.frame .....	34
10.4	pkm.frame.determines.format .....	34
10.5	pkm.tb.prefix[n] .....	35
10.6	pkm.tb.log .....	35
10.7	pkm.tb.update.macro.prev .....	35
<b>11</b>	<b>defaults.m .....</b>	<b>36</b>
11.1	lpmi_c_plot_set_plot_transformation_generic .....	36
11.2	lpmi_c_plot_set_plot_transformation_a4 (_a3, _a2 etc.) .....	36
11.3	lpmi_plot_g_line_scale .....	36
11.4	lpmi_plot_delete_elem .....	36
<b>12</b>	<b>DXFDWG.con .....</b>	<b>37</b>
12.1	dxfdwg.config.file .....	37
<b>13</b>	<b>Fehlersuche (Troubleshooting) .....</b>	<b>38</b>
13.1	me.transmode .....	38
13.2	debug.exit .....	38
13.3	debug.process.start .....	38
<b>14</b>	<b>Konfigurationsbeispiele .....</b>	<b>39</b>

## 1 Installation

### 1.1 Systemvoraussetzungen

COP Dok unterstützt die folgenden Betriebssysteme:

Windows 7 32 Bit

Windows 7 64 Bit

Windows 8.1 32 Bit

Windows 8.1 64 Bit

Auf den Systemen auf denen COP Dok eingesetzt wird, **muss** Creo Elements/Direct Drafting (ehemals CoCreate Drafting) oder Creo Elements/Direct 2D Access (ehemals CoCreate 2D Access) ab Version 17 installiert sein. Innerhalb des Netzwerkes muss ein Creo Elements/Direct Lizenzserver (MEIs) installiert, konfiguriert und verfügbar sein. COP Dok unterstützt nur die 32 Bit Versionen von Creo Elements/Direct Drafting oder Creo Elements/Direct 2D Access.

Microsoft .NET Framework (ab Version 2.0) muss installiert sein. Sie können .NET bei Microsoft herunterladen.

#### Optional (Pixeldateien)

Falls sie mit COP Dok Zeichnungen in Pixeldateien, wie TIFF, JPEG, GIF etc. konvertieren möchten, muss Ghostscript auf ihrem System installiert sein. Ghostscript kann bei <http://sourceforge.net/projects/ghostscript> bezogen werden.

#### Optional (3D Modelle)

Falls sie 3D Modelle, die sie Creo Elements/Direct Modeling (ehemals CoCreate Modeling) haben, müssen sie Creo Elements/Direct Modeling (ab Version 18) installieren. COP Dok unterstützt keine 3D Modell Konversion auf 32 Bit Windowsversionen.

### 1.2 Hinweis zu den Lizenzen

COP Dok enthält keine Creo Elements/Direct Drafting, Creo Elements/Direct 2D Access, Creo Elements/Direct Lizenzserver Creo Elements/Direct Modeling Lizenzen. Diese Lizenzen müssen Sie separat erwerben.

### 1.3 Vorbereitung

#### 1.3.1 Informationen beschaffen

Während der Installation werden Sie nach folgenden Informationen gefragt:

- Installationsverzeichnis von COP Dok (Standard: C:\Programme\cop\dok)
- Installationsverzeichnis von Creo Elements/Direct Drafting oder Creo Elements/Direct 2D Access
- Name oder IP Adresse des Lizenzservers (MEIs)
- Optional: Installationsverzeichnis von Ghostscript
- COP Dok Demo- oder Nodelocked Lizenz

#### 1.3.2 Lizenz in Creo Elements/Direct Lizenzserver (MEIs) eintragen

COP Dok beinhaltet einen Passwortschutz. Tragen Sie die COP Dok Lizenz (Zertifikatsnummer und Passwort) in die Konfigurationsdatei Ihres Creo Elements/Direct Lizenzservers (Mels) ein.

Falls Sie COP Dok für Testzwecke oder eine arbeitsplatzbezogenen (nodelocked) Lizenz installieren, können Sie das von uns gelieferte Passwort während der Installation angeben. In diesem Fall müssen Sie keinen Eintrag in die Konfigurationsdatei Ihres Creo Elements/Direct Lizenzservers (Mels) machen.

## 1.4 Installation durchführen

Starten Sie das Installationsprogramm copsetup.exe und folgen Sie den Instruktionen. Klicken Sie jeweils auf [Weiter >] um zum nächsten Dialog zu gelangen.

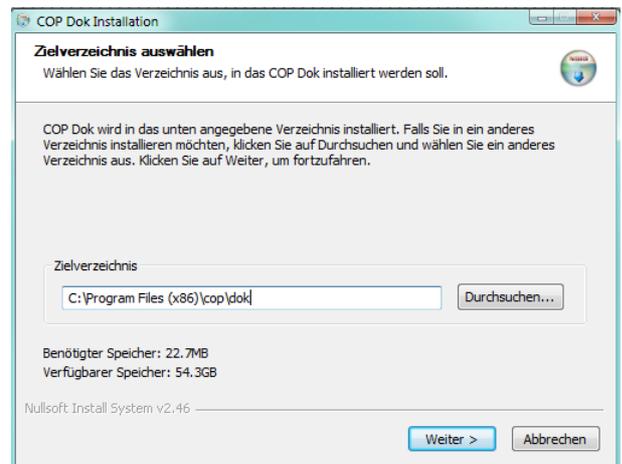
Wählen Sie das Installationsverzeichnis für COP Dok aus.

Das Standardverzeichnis ist:

C:\Programme\cop\dok oder

C:\Program Files (x86)\cop\dok (64 Bit Windows).

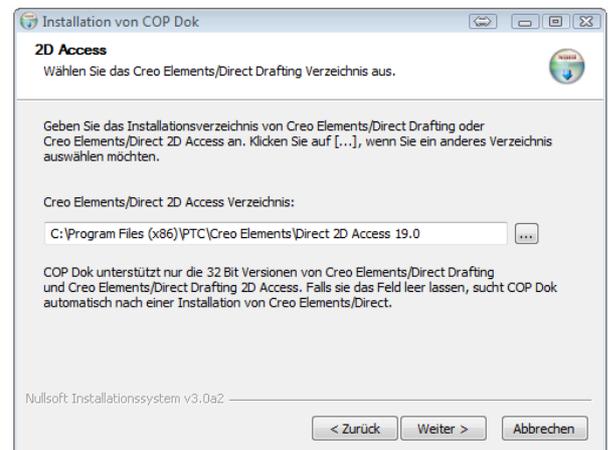
Falls Sie COP Dok in ein anderes Verzeichnis installieren möchten, dann klicken Sie auf [Durchsuchen ..] und wählen Sie das gewünschte Verzeichnis aus.



Wählen Sie das Installationsverzeichnis von Creo Elements/Direct Drafting oder Creo Elements/Direct 2D Access aus.

Falls Sie die Creo Elements/Direct Software in einem anderen Verzeichnis installieren haben, klicken Sie auf [ ... ] und wählen Sie das Verzeichnis aus.

Sie können das Feld auch leer lassen. In diesem Fall wird COP Dok das Installationsverzeichnis von Creo Elements/Direct Drafting oder Creo Elements/Direct 2D Access selbständig bestimmen.



### Hinweis

COP Dok unterstützt nur die 32 Bit Versionen von Creo Elements/Direct Drafting oder Creo Elements/Direct 2D Access.

Der Name des Installationsverzeichnisses von Creo Elements/Direct 2D Access wird vom Installationsprogramm in die Datei **lpmi.ini** eingetragen. Die Datei lpmi.ini befindet sich nach der Installation im Installationsverzeichnis von COP Dok.

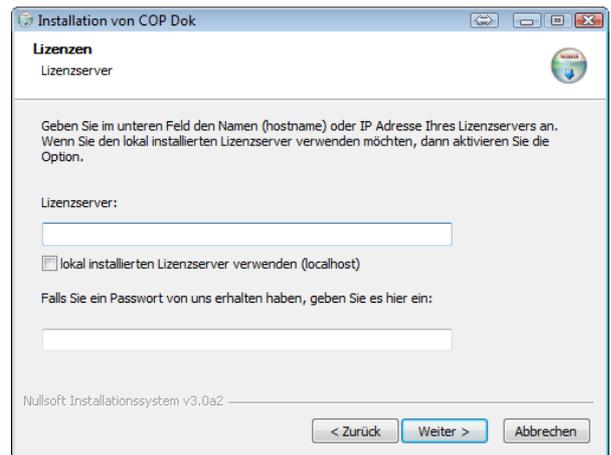
Geben Sie den Computernamen (Hostname oder IP Adresse) ihres Lizenzservers an.

Falls Sie einen lokal installierten Lizenzserver (localhost) verwenden möchten, aktivieren Sie die entsprechende Option.

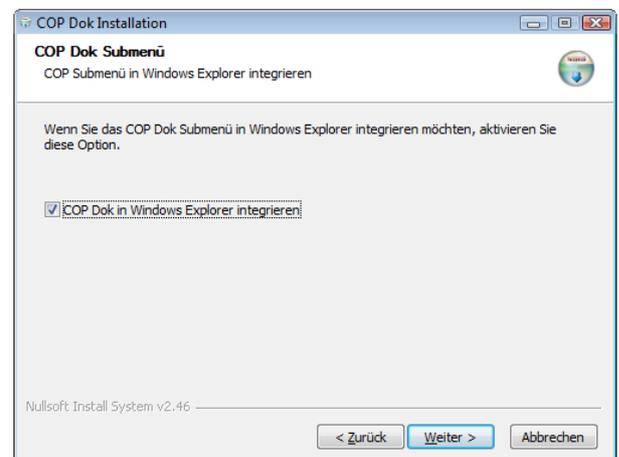
Der Name des Lizenzservers wird in die Konfigurationsdatei lpmi.ini (Schlüssel: license.server) gespeichert.

Falls Sie ein temporäres oder nodelocked Passwort von uns erhalten haben, dann geben Sie es in das entsprechende Feld ein.

Das temporäre Passwort wird in die Konfigurationsdatei lpmi.ini (Schlüssel: license.key) gespeichert.



Falls Sie das Rechte-Maustasten-Menü "COP Dok" in den Windows Explorer integrieren möchten, dann aktivieren Sie die Option (  ).



### Hinweis

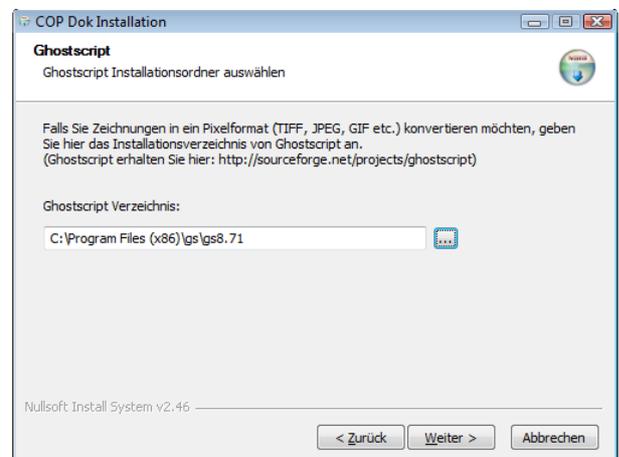
Falls Sie COP Dok bereits auf dem System installiert haben und eine Neuinstallation oder einen Update des Produktes durchführen, müssen Sie alle geöffneten Windows Explorer schliessen, bevor Sie die Installation mit [Installieren] starten. Dies ist zwingend notwendig, damit alle Softwarekomponenten vollständig installiert werden können.

Falls Sie mit COP Dok Ihre Zeichnungen in Pixeldateien, wie TIFF, JPEG, GIF etc. konvertieren möchten, dann muss Ghostscript auf Ihrem System installiert sein. Geben Sie hier das Installationsverzeichnis von Ghostscript (inkl. Version) an.

Beispiel:

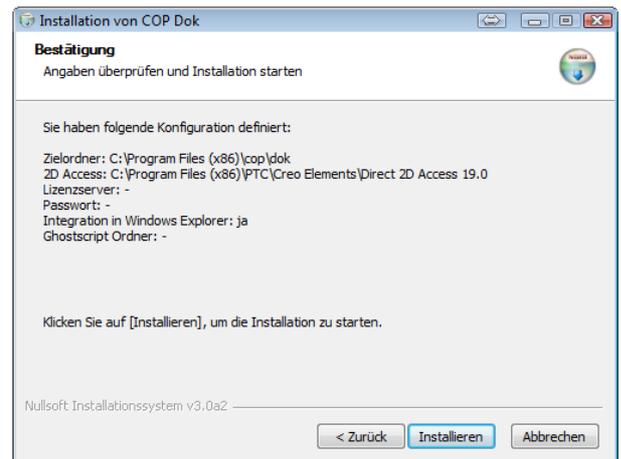
C:\Program Files (x86)\gs\gs8.71

Das Installationsverzeichnis von Ghostscript wird in die Konfigurationsdatei lpmi.ini (Schlüssel: converter.gs.dir) gespeichert.



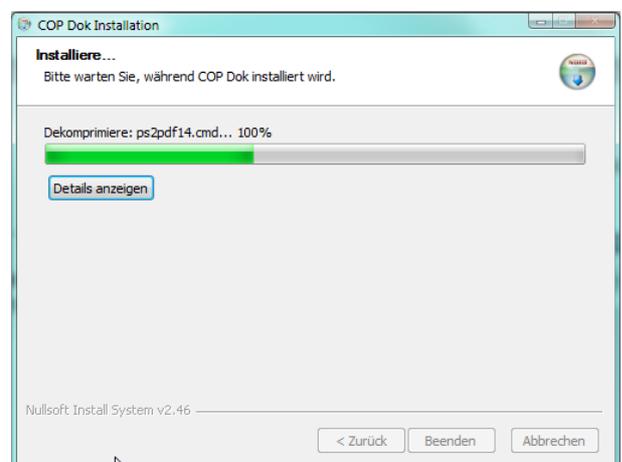
Sie können Ghostscript bei <http://sourceforge.net/projects/ghostscript> beziehen.

Überprüfen Sie Ihre Angaben und starten Sie die Installation mit [Installieren].



Die Installation startet.

Klicken Sie auf [Details anzeigen], falls Sie mehr Informationen zum Installationsfortschritt sehen möchten.



Klicken Sie auf [ OK ] um den Dialog zu schliessen.

## 1.5 COP Dok Konfiguration

Nach der Installation muss COP Dok den Gegebenheiten (Drucker etc.) Ihrer CAD Umgebung angepasst werden. Ausführliche Informationen zur Konfiguration finden Sie im Kapitel "Konfiguration".

## 2 Konfiguration

COP Dok können sie vielfältigen Aufgaben und Ansprüchen entsprechend anpassen und konfigurieren.

### 2.1 Konfigurationsdateien

Die Konfigurationen und Kundenanpassungen von COP Dok erfolgen in den folgenden Dateien:

- lpmi.ini

Die Voreinstellungen (Standards) befinden sich in diesen Dateien:

- defaults.ini
- defaults.m

### 2.2 lpmi.ini

Die Konfigurationsdatei lpmi.ini befindet im COP Dok Installationsordner (i.R. %PROGRAMFILES%\cop\dok).

In dieser Datei werden die folgenden Konfigurationen festgelegt:

- Allgemeine Konfigurationen (Installationspfad der Creo Elements/Direct Software, LOG Dateien etc.)
- Druckerkonfigurationen
- Konverter Konfigurationen

Die Datei kann mit einem herkömmlichen Texteditor (zum Beispiel Notepad) erstellt und geändert werden.

### 2.3 Konventionen für lpmi.ini und default.m

- Die Formate der Dateien lpmi.ini und defaults.ini entsprechen dem Windows INI Format.
- Eine Konfiguration wird als Schlüssel/Wert Paar der Form "Schlüssel=Wert" in lpmi.ini eingetragen.
- Eine Zeile in lpmi.ini enthält nur ein Schlüssel/Wert Paar.
- Bei Konfigurationen, die eine Gruppe definieren (zum Beispiel eine Gruppe von Druckern oder Konvertern) werden zur Unterscheidung die Schlüssel mit einem Index ergänzt, der in eckiger Klammer angegeben wird.

Beispiele:

```
printer[0].name = HP LaserJet
printer[1].name = HP DesignJet
converter[pdf].name = PDF
converter[dxf].name = DXF
```

- Für die Konfigurationen printer[] und converter[] sind beliebige Zeichenketten als Index möglich. Für alle anderen Konfigurationen werden Zahlen 0, 1, 2 ... verwendet. Der Index beginnt bei 0 und wird fortlaufend um 1 erhöht.
- Als Wert eines Schlüssel/Wert Paares gilt die Zeichenkette nach dem ersten Gleichheitszeichen (=). Die Leerzeichen vor und nach der Zeichenkette werden nicht berücksichtigt. Die nachfolgenden Konfigurationen sind somit identisch:

```
converter[pdf].name=PDF
converter[pdf].name = PDF
```

- Ein Wert einer Konfiguration kann einen Bezug auf Werte anderer Konfigurationen enthalten. Ein solcher Bezug wird mit  $\${Schlüssel}$  definiert.

Beispiel:

```
me.dir = C:\Program Files (x86)\PTC\Creo Elements\Direct 2D Access 18.1
```

```
me.customize.file= ${me.dir}\customize.m
```

In diesem Fall ist der Wert von me.customize.file zur Laufzeit gleich "C:\Program Files (x86)\PTC\Creo Elements\Direct 2D Access 18.1\customize.m".

- Ein Wert einer Konfiguration kann einen Bezug auf eine Umgebungsvariable (Systemvariable) von Windows beinhalten. Ein Bezug auf eine Umgebungsvariable wird mit `%VARIABLE%` definiert.

Beispiel:

```
system.tmp.dir=%TEMP%\%USERNAME%
```

In diesem Fall ersetzt COP Dok die Zeichenketten `%TEMP%` und `%USERNAME%` durch den Wert der gleichnamigen Umgebungsvariablen.

(Beispiel: system.tmp.dir = C:\temp\Administrator)

- Die Datei defaults.m ist eine Drafting Makrodatei und folgt dem Syntax der Drafting Makrosprache. Informationen über diese Makrosprache sind der Creo Elements/Direct Drafting Dokumentation zu entnehmen.

### 3 Allgemeine und System Konfigurationen in lpmi.ini

#### 3.1 loadfile[n]

Dieser Schlüssel definiert weitere INI Konfigurationsdateien, die von COP Dok geladen werden (sofern die Dateien vorhanden sind). Diese Dateien enthalten zum Beispiel benutzer- oder systemabhängige Konfigurationen.

COP Dok lädt zuerst alle Schlüssel aus lpmi.ini und dann die Schlüssel der Dateien, die in loadfile[n] definiert sind. Die Schlüssel in den loadfile[n] Dateien überschreiben somit die Schlüssel aus lpmi.ini.

Der Index n von loadfile[n] beginnt bei 0 und muss fortlaufend um 1 erhöht werden.

##### Beispiele

loadfile[0] = \\serverxyz\hostsdata\%COMPUTERNAME%.ini

Mit dieser Konfiguration lädt COP Dok eine vom Computernamen (%COMPUTERNAME%) abhängige INI Datei.

loadfile[1] = \\serverxyz\usersdata\%USERNAME%.ini

Mit dieser Konfiguration lädt COP Dok eine vom Benutzernamen (%USERNAME%) abhängige INI Datei.

#### 3.2 log.config

COP Dok schreibt während der Ausführung diverse Informationen in LOG Dateien. COP Dok speichert die LOG Dateien im Ordner %TEMP%\lpmilog. U.a. werden auch alle Konfigurationen (aus lpmi.ini u.a.) in die LOG Dateien geschrieben. Bei sehr umfangreichen Konfigurationen, kann dies die Ausführungsdauer merklich verlängern. Falls log.config = 0 ist, schreibt COP Dok die Konfigurationen nicht in die LOG Dateien.

##### Standardwert

1

##### Beispiel

log.config = 0

#### 3.3 progressbar.start

Während der Ausführung zeigt COP Dok einen Fortschrittsbalken an.



Falls der Schlüssel progressbar.start = 0 ist, zeigt COP Dok den Fortschrittsbalken nicht an.

##### Standardwert

1

##### Beispiel

progressbar.start= 0

### 3.4 **system.env.set.Systemvariablen**

Mit dem Schlüssel können Umgebungsvariablen (Systemvariablen) definiert werden. Damit können bei Bedarf bestimmte Systemvariablen geändert oder neu definiert werden.

#### **Beispiel**

system.env.set.DXFDIR = C:\%PROGRAMFILES%\mydxfwg

Der Name der Variable ist im Schlüssel zu definieren (im Beispiel DXFDIR).

### 3.5 **system.tmp.dir**

Mit diesem Schlüssel kann der Ordner festgelegt werden, in den COP Dok temporärer Dateien ablegt. COP Dok erstellt in diesem Ordner zwei weitere Ordner (Ipmi tmp und Ipmilog). Der angegebene Ordner muss existieren.

#### **Standardwert**

%TEMP%

#### **Beispiel**

system.tmp.dir=%TEMP%\%USERNAME%

#### **Siehe auch**

defaults.ini

## 4 **Creo Elements/Direct Drafting und 2D Access Konfigurationen**

### 4.1 **hiddenline.enable**

Falls die Zeichnung sog. Hidden Line Elemente enthält, bereitet COP Dok vor der Ausgabe der Zeichnung mit dem Drafting Befehl HL\_GENERATE\_HIDDEN die versteckten Kanten auf. Mit diesem Schlüssel kann die Aufbereitung der versteckten Kanten global für alle Konverter Konfigurationen ein- oder ausgeschaltet werden.

#### **Gültige Werte**

1 = versteckte Kanten aufbereiten

0 = versteckte Kanten nicht aufbereiten

#### **Standardwert**

0

#### **Beispiel**

hiddenline.enable = 1

#### **Siehe auch**

converter[.].hiddenline.enable

### 4.2 **me.dir**

Dieser Schlüssel definiert das Installationsverzeichnis von Creo Elements/Direct Drafting oder Creo Elements/Direct 2D Access. Der Schlüsselwert wird automatisch bei der Installation erstellt.

#### **Beispiele**

me.dir = C:\Program Files (x86)\PTC\Creo Elements\Direct 2D Access 18.1

#### **Hinweis:**

Falls der Schlüssel me.dir nicht definiert oder leer ist, sucht COP Dok das Installationsverzeichnis von Creo Elements/Direct automatisch.

COP Dok sucht in dieser Reihenfolge nach Creo Elements/Direct:

%PROGRAMFILES%\PTC\Creo Elements\Direct Drafting 19.0

%PROGRAMFILES%\PTC\Creo Elements\Direct 2D Access 19.0

%PROGRAMFILES%\PTC\Creo Elements\Direct Drafting 18.1  
 %PROGRAMFILES%\PTC\Creo Elements\Direct 2D Access 18.1  
 %PROGRAMFILES%\PTC\Creo Elements\Direct Drafting 18.0  
 %PROGRAMFILES%\PTC\Creo Elements\Direct 2D Access 18.0  
 %PROGRAMFILES%\PTC\Creo Elements\Direct Drafting 17.0  
 %PROGRAMFILES%\PTC\Creo Elements\Direct 2D Access 17.0  
 %PROGRAMFILES%\CoCreate\CoCreate Drafting 17.0  
 %PROGRAMFILES%\CoCreate\CoCreate 2D Access 17.0

### 4.3 **me.customize.file**

Dieser Schlüssel definiert eine Drafting Makrodatei, die während dem Start von COP Dok eingelesen wird. Diese Datei kann Drafting Makros enthalten, die von COP Dok entsprechend der Konfiguration ausgeführt werden.

#### **Beispiele**

me.customize.file = \${me.dir}\customize.m

#### **Siehe auch**

defaults.m  
 macro.load.post  
 printer[*n*].macro.transformation

### 4.4 **macro.load.post**

Dieser Schlüssel definiert den Namen eines Drafting Makros, welches unmittelbar aufgerufen wird, nachdem eine Zeichnung in Creo Elements/Direct Drafting oder 2D Access geladen wurde. Damit können zum Beispiel Geometrieelemente gelöscht werden, die nicht gedruckt werden sollen (Rahmenbegrenzer etc.).

Das Makro muss während dem Startup von Creo Elements/Direct Drafting oder Creo Elements/Direct 2D Access geladen werden

#### **Siehe auch**

me.customize.file

### 4.5 **me.start.drafting**

COP Dok startet immer Creo Elements/Direct 2D Access. Ist der Schlüssel me.start.drafting = 1, dann startet Creo Elements/Direct Drafting.

#### **Standardwert**

0

#### **Beispiel**

me.start.drafting = 1

## 5 Konfigurationen für Pixmap in Zeichnungen

In einer 2D Zeichnungen können Pixmap (Grafiken) eingebettet sein (zum Beispiel gerenderte 3D Ansichten). COP Dok gibt diese Grafiken aus.

### 5.1 **pixmap.quality**

Der Schlüssel pixmap.quality definiert die Qualität der ausgegeben Grafik.

#### **Gültige Werte**

best, low

#### **Standardwert**

best

#### **Beispiel**

pixmap.quality = low

### 5.2 **pixmap.color**

Der Schlüssel pixmap.color definiert die Farbe der ausgegeben Grafik.

#### **Gültige Werte**

color = farbige Ausgabe

bw = Schwarzweis

#### **Standardwert**

color

#### **Beispiel**

pixmap.color = bw

## 6 Creo Elements/Direct 3D Access und Modeling Konfiguration

Für die Konversion von 3D Modellen muss Creo Elements/Direct 3D Access oder Modeling installiert sein.

### Hinweis

COP Dok unterstützt nur die 64 Bit Versionen von Creo Elements/Direct 3D Access und Modeling.

### 6.1 sd.dir

Der Schlüssel sd.dir definiert das Installationsverzeichnis von Creo Elements/Direct 3D Access oder Modeling. Falls sie 3D Modelle konvertieren möchten, muss dieser Schlüssel konfiguriert sein.

#### Standardwert

-

#### Beispiel

sd.dir = C:\Program Files\PTC\Creo Elements\Direct 3D Access 19.0

### Hinweis

Falls sie 3D Modelle konvertieren möchten, muss dieser Schlüssel konfiguriert sein.

### 6.2 sd.application

Der Schlüssel sd.application definiert, ob COP Dok Creo Elements/Direct 3D Access oder Modeling für die Konversion von 3D Modellen verwenden soll. Falls sie 3D Modelle konvertieren möchten, muss dieser Schlüssel konfiguriert sein.

#### Gültige Werte

Wert	Beschreibung
3daccess	COP Dok verwendet Creo Elements/Direct 3D Access
modeling	COP Dok verwendet Creo Elements/Direct 3D Modeling

#### Standardwert

-

#### Beispiel

sd.application = 3daccess

### 6.3 sd.application.args

Mit diesem Schlüssel kann während dem Start ein Argument an Creo Elements/Direct 3D Access oder Modeling übergeben werden.

#### Standardwert

-

#### Beispiel

sd.application.args = -v

#### 6.4 **sd.application.kill.process.onstart**

COP Dok prüft vor dem Start von Creo Elements/Direct 3D Access oder Modeling, ob bereits eine Instanz läuft. Ist dies nicht der Fall, startet COP Dok eine neue Instanz. `sd.application.kill.process.onstart = 1`, dass beendet COP Dok vor dem Start einer neuen Creo Elements/Direct 3D Access oder Modeling Instanz sämtliche laufenden Instanzen.

**Hinweis**

Alle bereits laufenden Instanzen von Creo Elements/Direct 3D Access oder Modeling werden beendet!

**Standardwert**

0

**Beispiel**

`sd.application.kill.process.onstart = 1`

**Siehe auch**

`sd.application.restart.if_not_responding`

#### 6.5 **sd.application.kill.process.all**

Falls COP Dok laufende Creo Elements/Direct 3D Access oder Modeling Instanzen beendet (siehe `sd.application.kill.process.onstart`, `sd.application.restart.always` oder `sd.application.restart.if_not_responding`), dann beendet COP Dok nur die Instanzen der mit `sd.application` definierten Programme.

Falls `sd.application.kill.process.all = 1` ist, dann beendet COP Dok alle Instanzen von Creo Elements/Direct 3D Access und Modeling.

**Hinweis**

Alle bereits laufenden Instanzen von Creo Elements/Direct 3D Access und Modeling werden beendet!

**Standardwert**

0

**Beispiel**

`sd.application.kill.process.all = 1`

**Siehe auch**

`sd.application.kill.process.onstart`

`sd.application.restart.always`

`sd.application.restart.if_not_responding`

## 6.6 **sd.application.restart.always**

COP Dok prüft vor dem Start von Creo Elements/Direct 3D Access oder Modeling, ob bereits eine Instanz läuft. Ist dies der Fall, dann startet COP Dok keine neue Instanz, sondern verwendet die aktuell laufenden Instanzen.

Ist `sd.application.restart.always = 1`, dann startet COP Dok immer eine neue Instanz der Programme.

### **Hinweis**

Eine bereits laufende Instanz wird automatisch beendet!

### **Standardwert:**

0

### **Beispiel**

`sd.application.restart.always = 1`

## 6.7 **sd.application.restart.if\_not\_responding**

COP Dok prüft vor dem Start von Creo Elements/Direct 3D Access oder Modeling, ob bereits eine Instanz läuft. Ist dies der Fall, dann startet COP Dok keine neue Instanz, sondern verwendet die aktuell laufenden Instanzen.

Ist `sd.application.restart.if_not_responding = 1`, dann startet COP Dok eine neue Instanz der Programme, falls diese nicht mehr auf Befehle von COP Dok reagieren.

### **Standardwert**

0

### **Beispiel**

`sd.application.restart.if_not_responding = 1`

### **Siehe auch**

`sd.application.maxwait.load`

`sd.application.kill.process.onstart`

## 6.8 **sd.application.maxwait.load**

COP Dok weist Creo Elements/Direct 3D Access oder Modeling an, ein 3D Modell zu laden und wartet eine bestimmte Zeit darauf. Falls Access oder Modeling nicht innerhalb der festgelegten Zeit reagiert, nimmt COP Dok an, dass ein Fehler aufgetreten ist. Mit dem Schlüssel `sd.application.maxwait.load` wird die Wartezeit in Sekunden festgelegt.

### **Standardwert**

120

### **Beispiel**

`sd.application.maxwait.load=300`

## 6.9 **sd.delete.all**

Der Schlüssel `sd.delete.all` definiert, ob COP Dok vor dem Laden eines 3D Modells in Creo Elements/Direct 3D Access oder Modeling alle bereits geladenen 3D Modelle löschen soll.

COP Dok prüft vor dem Start von Creo Elements/Direct 3D Access oder Modeling, ob diese Programme bereits laufen. Ist dies der Fall, dann startet COP Dok keine neue Instanz der Programme, sondern verwendet die aktuell laufenden Programme.

### **Hinweis**

Falls `sd.delete.all = 1` gesetzt ist, werden bereits geladene Modelle entfernt!

### **Standardwert**

0

### **Beispiel**

`sd.delete.all = 1`

## 7 Druckerkonfigurationen

Auf einem System sind in der Regel mehrere Drucker konfiguriert. Die Druckerkonfigurationen in lpmi.ini werden mit einem Index in eckigen Klammern unterschieden.

Beispiele:

```
printer[laserjet]
printer[designjet]
printer[0]
printer[1]
```

Für printer[**n**] Konfigurationen können für **n** beliebige Zeichenketten eingesetzt werden. Für alle anderen Konfigurationen mit Indexe muss der Index mit 0 beginnen und fortlaufend um 1 erhöht werden.

Beispiele:

```
printer[designjet].format[0] = A0
printer[designjet].format[1] = A1
printer[designjet].format[2] = A3
```

### 7.1 printer[*n*].name

Dieser Schlüssel enthält den frei wählbaren Namen einer Druckerkonfiguration. Auf diesen Namen beziehen sich weitere Konfigurationen (zum Beispiel printer.default). Dieser Name erscheint in der Drucker Auswahlliste, wenn eine 2D Zeichnung ausgedruckt wird.

Dieser Schlüssel muss für jede Druckerkonfiguration definiert sein.

#### Beispiel

```
printer[laserjet].name = HP LaserJet 1.Stock
printer[designjet].name = HP DesignJet
```

### 7.2 printer[*n*].system.name

Dieser Schlüssel enthält den Namen des Druckers, wie dieser im Windows Betriebssystem konfiguriert ist. Der Name muss exakt der Windows Konfiguration entsprechen (siehe Start/Einstellungen/Drucker und Faxgeräte).

Dieser Schlüssel muss für jede Druckerkonfiguration definiert sein.

#### Beispiel

```
printer[laserjet].system.name = HP LaserJet 5040 Series PCL
printer[designjet].system.name = \\printerserver\designjet01
```

### 7.3 printer[*n*].format[*i*]

Dieser Schlüssel definiert die vom Drucker unterstützten Ausgabeformate.

Die einzelnen Formate werden mit einem Index 0,1,2, etc. unterschieden.

Dieser Schlüssel muss für jede Druckerkonfiguration definiert sein.

#### Gültige Werte

A0, A1, A2, A3, A4

#### Beispiel

```
printer[laserjet].format[0] = A4
printer[laserjet].format[1] = A3
printer[designjet].format[0] = A0
printer[designjet].format[1] = A1
printer[designjet].format[2] = A2
```

#### 7.4 **printer[n].format[i].system.format**

Falls im Windows Druckertreiber die Formate anders als mit A0, A1 ... A4 bezeichnet sind, müssen die Formatbezeichnungen mit diesem Schlüssel definiert werden.

**Beispiel:**

```
printer[laserjet].format[0] = A4
```

```
printer[laserjet].format[0].system.format = A4 (210 x 297 mm)
```

#### 7.5 **printer[n].format[i].width**

Mit diesem Schlüssel kann die Breite des Druckbereiches des Formats format[i] definiert werden.

**Standardwert**

Es gelten die normierten Breiten der entsprechenden Formate (A0..A4).

**Beispiel**

```
printer[0].format[0].width = 205
```

#### 7.6 **printer[n].format[i].height**

Mit diesem Schlüssel kann die Höhe des Druckbereiches des Formats format[i] definiert werden.

**Standardwert**

Es gelten die normierten Höhen der entsprechenden Formate (A0..A4).

**Beispiel**

```
printer[0].format[0].height = 290
```

#### 7.7 **printer[n].format[i].xoffset**

Dieser Schlüssel definiert eine Verschiebung in X-Richtung der Zeichnung auf dem Papier.

**Standardwert**

0

**Beispiel**

```
printer[0].format[0].xoffset = 5
```

**Siehe auch**

Drafting Befehl PLOT

#### 7.8 **printer[n].format[i].yoffset**

Dieser Schlüssel definiert eine Verschiebung in Y-Richtung der Zeichnung auf dem Papier.

**Standardwert**

0

**Beispiel**

```
printer[0].format[0].yoffset = 5
```

**Siehe auch**

Drafting Befehl PLOT

**7.9 printer[n].format[i].scale**

Dieser Schlüssel definiert eine Skalierung für das Format format[i]. Die Zeichnung wird vor der Ausgabe mit diesem Wert skaliert.

**Standardwert**

1

**Beispiel**

```
printer[0].format[0].scale = 0.99
```

**7.10 printer[n].format[i].orientation**

Mit diesem Schlüssel kann definiert werden, ob eine Zeichnung im Querformat (landscape) oder Hochformat (portrait) ausgegeben werden soll.

**Gültige Werte**

landscape (= Querformat)

portrait (= Hochformat)

**Standardwert**

landscape

**Beispiel**

```
printer[0].format[0] = portrait
```

**7.11 printer[n].center**

Dieser Schlüssel gibt an, ob die Zeichnung auf dem Papier zentriert (Wert = 1) werden soll.

**Standardwert**

1

**Beispiel**

```
printer[0].center = 0
```

**Siehe auch**

Drafting Befehl PLOT\_ CENTER

**7.12 printer[n].rotate**

Dieser Schlüssel definiert den Drehwinkel der Zeichnung auf dem Papier (siehe auch Drafting Befehl PLOT).

**Standardwert**

0

**Beispiel**

```
printer[0].rotate = 90
```

**7.13 printer[n].macro.transformation**

Dieser Schlüssel definiert ein Drafting Makro, welches die Plot-Transformation (Stifte, Strichstärken etc.) ausführt. Dieses Makro wird vor dem Drucken aufgerufen. Das Makro muss während dem Startup von OSD Drafting/2D Access geladen werden.

**Standardwert**

lpmi\_plot\_set\_plot\_transformation\_generic

Das Standardmakro ist in der Datei defaults.m definiert.

**Beispiel**

```
printer[0].macro.transformation = black_and_white_transformation
```

**Siehe auch**

me.customize.file

**7.14 printer[n].macro.transformation.aN**

Die Zeichenkette aN steht für a0, a1, a2, a3, a4. Dieser Schlüssel definiert ein Drafting Makro, welches die Plot-Transformation (Stifte, Strichstärken etc.) ausführt. Dieses Makro wird in Abhängigkeit zum Ausgabeformat vor dem Drucken aufgerufen. Das Makro muss während dem Startup von OSD Drafting/2D Access geladen werden (siehe defaults.m).

**Standardwert**

lpmi\_plot\_set\_plot\_transformation\_generic\_aX

**Beispiel**

printer[0].macro.transformation.a4 = black\_white\_transform\_a4

printer[0].macro.transformation.a3 = black\_white\_transform\_a3

**Siehe auch**

defaults.m

**7.15 printer.default**

Dieser Schlüssel definiert den Standarddrucker. Dieser Drucker wird ausgewählt, wenn der Benutzer in der Druckerauswahl <Automatisch> auswählt. Der Name bezieht sich auf die Angaben in printer[n].name.

**Beispiel**

printer.default = LaserJet

**7.16 printer.default.aN**

Die Zeichenkette aN steht für a0, a1, a2, a3, a4. Dieser Schlüssel definiert den Standarddrucker für ein bestimmtes Zeichnungsformat. Der Name bezieht sich auf die Angaben in printer[n].name. Falls für das aktuelle Zeichnungsformat kein passender Drucker definiert ist, wird die Zeichnung auf den Standarddrucker (printer.default) ausgegeben.

**Beispiel**

printer.default.A0 = DesignJet

**7.17 printer.macro.load.post**

Dieser Schlüssel definiert ein Makro, welches unmittelbar aufgerufen wird, nachdem eine Zeichnung in Drafting/2D Access geladen wurde. Damit können zum Beispiel Geometrien gelöscht werden, die nicht gedruckt werden sollen (Rahmenbegrenzer etc.).

Das Makro muss während dem Startup von Creo Elements/Direct Drafting/2D Access geladen werden (siehe customize.m).

**Standardwert**

lpmi\_plot\_delete\_elem

**Beispiel**

printer.macro.load.post = do\_my\_macro

**Siehe auch**

macro.load.post

defaults.m

**7.18 printer[n].enable**

Mit diesem Schlüssel können einzelne Druckerkonfigurationen gezielt ein- oder ausgeschaltet werden. Dies ist nützlich um zum Beispiel für einzelne Arbeitsstationen oder Anwender nicht verfügbare Drucker zu deaktivieren.

**Standardwert**

1

**Beispiel**

```
printer[laserjet].enable = 0
```

**Siehe auch**

```
loadfile[n]
```

## 8 Konverter Konfigurationen

In lpmi.ini können mehrere Konverter konfiguriert werden. Die Konverter Konfigurationen in lpmi.ini werden mit einem Index in eckigen Klammern unterschieden.

Beispiele:

converter[pdf]

converter[dxfl

converter [0]

converter [1]

### 8.1 converter[n].name

Dieser Schlüssel enthält den frei wählbaren Namen eines Konverters. Dieser Name erscheint in der Auswahlliste, wenn eine Zeichnung konvertiert wird.

#### Beispiel

converter[pdf].name = PDF

converter[dxfl.name = DXF

### 8.2 converter[n].type

Dieser Schlüssel definiert das Ausgabeformat.

#### Gültige Werte

Wert	Ausgabeformat
creoview	PTC Creo View (nur 3D)
dwg	DWG
dxfl	DXF
edrawing	eDrawing (nur 3D)
gif	GIF Grafik
hppl	HPGL
igs	IGES
jpg	JPEG Grafik
multiple	mehrere Konversionen gleichzeitig ausführen (siehe delegate[])
pdf	PDF
pkg	PKD (Creo Elements/Direct Format, nur 3D)
sat	SAT (nur 3D)
step	STEP (nur 3D)
stl	STL (nur 3D)
svg	SVG
tif	TIFF Grafik
vrml	VRML

#### Beispiel

converter[pdf].type = pdf

#### Hinweise

Falls Sie eine Pixelformat (Grafik), wie TIFF, JPEG, GIF etc. als Ausgabeformat konfigurieren, dann muss Ghostscript auf Ihrem System installiert sein (siehe Installation).

Für bestimmte Ausgabeformate sind u.U. zusätzliche Lizenzen notwendig (zum Beispiel PDF Export aus Creo Elements/Modeling). Diese Lizenzen sind nicht Bestandteil von COP Dok.

### 8.3 **converter[n].default**

Mit diesem Schlüssel kann die Konfiguration festgelegt werden, die angewendet wird, falls der Benutzer im COP Dok Sub Menü "in PDF", "in DXF" oder "in DWG" auswählt. Wenn der Benutzer "in PDF" auswählt, sucht COP Dok nach einer Konfiguration mit "converter[n].type = pdf" und "converter[n].default=1". Findet COP Dok keine Konfiguration mit "converter[n].default=1", wählt es die erste Konfiguration mit "converter[n].type = pdf" aus. Ist keine Konfiguration mit "converter[n].type = pdf" vorhanden, werden Standardwerte für die PDF Generierung verwendet.

Das gleiche Vorgehen gilt für "in DXF" (converter[n].type = dxf) und "in DWG" (converter[n].type = dwg).

#### **Beispiel**

converter[pdf].default = 1

### 8.4 **converter[n].file.extension**

Dieser Schlüssel definiert die Dateiendung, die für die Ausgabedatei verwendet wird.

#### **Standardwert**

<b>Ausgabeformat</b>	<b>Endung</b>
Creo View	pvz
DWG	dwg
DXF	dxf
eDrawing	easm
GIF Grafik	gif
HPGL	hpg
IGES	igs
JPEG Grafik	jpg
PDF	pdf
PKG	pkg
SAT	sat
STL	stl
SVG	svg
STP	stp
TIFF Grafik	tif
VRML	vrml

#### **Beispiel**

converter[hpgl].file.extension = hpgl

**8.5 converter[n].macro.transformation**

Dieser Schlüssel definiert ein Drafting Makro, welches die Plot-Transformation (Stifte, Strichstärken etc.) ausführt. Dieses Makro wird unmittelbar vor der Ausgabe in das Zielformat aufgerufen. Das Makro muss während dem Startup von OSD Drafting/2D Access geladen werden.

**Standardwert**

lpmi\_plot\_set\_plot\_transformation\_generic

Dieses Standardmakro ist in der Datei defaults.m definiert.

**Beispiel**

converter[pdf].macro.transformation = pdf\_black\_and\_white\_transformation

**Siehe auch**

defaults.m

**8.6 converter[n].macro.transformation.aN**

Die Zeichenkette aN steht für a0, a1, a2, a3, a4. Dieser Schlüssel definiert ein Drafting Makro, welches die Plot-Transformation (Stifte, Strichstärken etc.) ausführt. Dieses Makro wird unmittelbar vor der Ausgabe in das Zielformat aufgerufen. Das Makro muss während dem Startup von OSD Drafting/2D Access geladen werden.

**Standardwert**

lpmi\_plot\_set\_plot\_transformation\_generic\_aN

**Beispiel**

converter[pdf].macro.transformation.a4 = black\_white\_transform\_a4

converter[pdf].macro.transformation.a3 = black\_white\_transform\_a3

**Siehe auch**

defaults.m

**8.7 converter[n].macro.load.post**

Dieser Schlüssel definiert ein Makro, welches unmittelbar aufgerufen wird, nachdem eine Zeichnung in Drafting/2D Access geladen wurde. Damit können zum Beispiel Geometrie gelöscht werden, die nicht ausgegeben werden sollen (Rahmenbegrenzer etc). Das Makro muss während dem Startup von Creo Elements/Direct Drafting/2D Access geladen werden.

**Standardwert**

lpmi\_plot\_delete\_elem

**Beispiel**

converter[pdf].macro.load.post = do\_my\_macro

**Siehe auch**

macro.load.post

defaults.m

**8.8 converter[n].macro.post**

Dieser Schlüssel definiert ein Makro, welches unmittelbar nach der Erstellung der Ausgabedatei ausgeführt wird. Dieses Makro wird für jede zu konvertierende Zeichnung ausgeführt. Das Makro muss während dem Startup von Creo Elements/Direct Drafting/2D Access geladen werden.

**Beispiel**

converter[pdf].macro.post = my\_makro

**Siehe auch**

defaults.m

## 8.9 Ausgabe

### 8.9.1 converter[n].output.file.name

Im Standard verwendet COP Dok den Namen der Quelldatei für den Namen der Ausgabedatei. Bei mehrblättrigen Zeichnungen hängt COP Dok die Blattnummer an den Namen an. Bei PKM Dateien hängt COP Dok die Blattnummer und Version an den Namen an. Mit dem Schlüssel converter[n].output.file.name kann der Ausgabedateiname angepasst werden.

Der Wert kann auch Attributnamen aus der Model Manager Export PKM Datei (Stammdatensattribute, Dokumentenattribute etc.) enthalten. Diese sind mit den Zeichen "<" und ">" zu umschliessen. Die Stammdatensattribute müssen mit "P\_" beginnen. Alle anderen Attribute beziehen sich auf Dokumentenattribute.

#### Standardwert

-

#### Beispiel

```
converter[pdf].output.file.name = <P_NAME>-<NAME>-<VERSION>-<P_DESCRIPTION>-<_CURRENT_SHEET_>
```

In diesem Beispiel verwendet COP Dok die Werte aus der PKM Datei. <P\_NAME> ist der Stammdatensname (oder Artikelnummer). <NAME> ist der Zeichnungsname (oder Zeichnungsnummer). <VERSION> ist die Version der Zeichnung. <P\_DESCRIPTION> ist die Beschreibung (oder Benennung) der Stammdatens. <\_CURRENT\_SHEET\_> ist die Blattnummer.

### 8.9.2 converter[n].output.file.name.blank.replaceby

Im Standard verwendet COP Dok den Namen der Quelldatei für den Namen der Ausgabedatei. Mit diesem Schlüssel kann ein Zeichen (oder Zeichenkette) definiert werden. COP Dok ersetzt dann alle Leerzeichen im Ausgabedateinamen durch diese Zeichen.

#### Standardwert

*keiner*

#### Beispiel

```
converter[pdf].output.file.name.blank.replaceby = _
```

#### Siehe auch

```
converter[n].output.file.name.blank.replaceby.no.duplicate
```

### 8.9.3 converter[n].output.file.name.blank.replaceby.no.duplicate

Falls dieser Schlüssel gleich 1 ist, dann ersetzt mehrfach hintereinander vorkommende Leerzeichen durch ein einzelnes Leerzeichen. Dieser Schlüssel kann nur mit dem Schlüssel output.file.name.blank.replaceby verwendet werden.

#### Standardwert

0

#### Beispiel

```
converter[pdf].output.file.name.blank.replaceby.no.duplicate = 1
```

#### Siehe auch

```
converter[n].output.file.name.blank.replaceby
```

#### 8.9.4 **converter[n].output.file.name.blank.remove**

Falls `output.file.name.blank.remove = 1` ist, dann entfernt COP Dok alle Leerzeichen aus dem Ausgabedateinamen.

**Standardwert**

0

**Beispiel**

`converter[pdf].output.file.name.blank.remove = 1`

**Siehe auch**

`converter[n].output.file.name.blank.replaceby`

#### 8.9.5 **converter[n].sheets.separator**

Bei der Ausgabe einer Zeichnung mit mehreren Blättern (i.R. eine Annotation Zeichnung) werden die Blätter in einzelne Dateien exportiert. Die Dateien werden mit der Blattnummer gekennzeichnet. Mit diesem Schlüssel wird das Trennzeichen zwischen Name und Blattnummer definiert.

**Standardwert**

Bindestrich -

**Beispiel**

`converter[pdf].sheets.separator = _`

#### 8.9.6 **converter[n].sheets.prefix**

Bei der Ausgabe einer Zeichnung mit mehreren Blättern werden die Blätter in einzelne Dateien exportiert. Mit diesem Schlüssel kann ein Präfix definiert werden, die dem Dateinamen vorangestellt wird.

**Beispiel**

`converter[pdf ].sheets.prefix = blatt-`

#### 8.9.7 **converter[n].sheets.postfix**

Bei der Ausgabe einer Zeichnung mit mehreren Blättern werden die Blätter in einzelne Dateien exportiert. Mit diesem Schlüssel kann ein Postfix definiert werden, die dem Dateinamen angehängt wird.

**Beispiel**

`converter[pdf ].sheets.postfix = -anno`

**8.9.8 converter[n].sheets.number.add.mode**

Bei der Ausgabe einer Zeichnung mit mehreren Blättern werden die Blätter in einzelne Dateien exportiert. Der Blattnummer wird dabei dem Dateinamen angehängt. Bei Zeichnungen, die mit Creo Elements/Direct Modeling (Annotation) erstellt wurden, wird in der Standardkonfiguration die Blattnummer immer angehängt, auch wenn die Zeichnung nur aus einem Blatt besteht. Mit diesem Schlüssel kann dieses Verhalten beeinflusst werden.

Wert	Beschreibung
0	COP Dok fügt die Blattnummer nur dann an den Dateinamen an, wenn die Zeichnung mindestens zwei Blätter enthält.
1	COP Dok fügt die Blattnummer immer an den Dateinamen an, auch wenn die Zeichnung nur aus einem Blatt besteht. Dies ist der Standardwert.

**Standardwert**

1

**Beispiel**

```
converter[pdf].sheets.number.add.mode = 0
```

**8.9.9 converter[n].destination.dir.name**

Mit diesem Schlüssel kann das Verzeichnis definiert, in das die Ausgabedateien gespeichert werden. Ist dieser Schlüssel nicht definiert, werden die Ausgabedateien in die gleichen Verzeichnisse gespeichert, in denen sich die Eingabedateien befinden oder in das Verzeichnis, welches der Anwender im COP Dok Dialog ausgewählt hat. Ist dieser Schlüssel definiert, wird das vom Benutzer ausgewählte Ausgabeverzeichnis nicht berücksichtigt.

**Beispiel**

```
converter[pdf].destination.dir.name = \\serverxyz\pdf
```

**8.9.10 converter[n].output.format**

Dieser Schlüssel definiert das Ausgabeformat der Zeichnung. Im Standard entspricht das Ausgabeformat dem Zeichnungsformat, resp. der Zeichnungsgröße (Breite x Höhe). Mit diesem Schlüssel kann das Ausgabeformat bewusst geändert werden, um zum Beispiel alle Zeichnungen immer auf A3 zu skalieren. Falls das Zeichnungsformat kleiner als das mit converter[n].output.format konfigurierte Format ist, dann entspricht das Ausgabeformat dem Zeichnungsformat.

Falls der Wert gleich "auto" ist, dann bestimmt COP Dok automatisch das zur Zeichnung passende Ausgabeformat A0...A4.

Ist dieser Schlüssel nicht definiert oder dessen Wert ist leer, dann entspricht das Ausgabeformat dem Zeichnungsformat, d.h. die Größe (oder Dimension) der Ausgabe stimmt mit der konvertierten Zeichnung überein.

**Standardwert**

keiner

**Gültige Werte**

A0,A1,A2,A3,A4

**Beispiel**

```
converter[0].output.format = A3
```

**Hinweise**

Dieser Schlüssel ist nur bei der Ausgabe einer PDF oder Grafikdatei (JPEG u.a.) aktiv.

**8.10 converter[n].delegate[m]**

Mit diesem Schlüssel können mehrere Konfigurationen zusammengefasst werden. COP Dok führt dann für eine Zeichnung alle mit delegate[] definierten Konversionen durch.

**Beispiel**

```
converter[pdf].name = PDF
converter[pdf].type = pdf
converter[pdf].default = 1
```

```
converter[dxfl].name = DXF
converter[dxfl].type = dxfl
```

```
converter[pdfdxfl].name = PDF & DXF
converter[pdfdxfl].type = multiple
converter[pdfdxfl].delegate[0] = pdf
converter[pdfdxfl].delegate[1] = dxfl
```

Bei diesem Beispiel erstellt COP Dok aus den ausgewählten Zeichnungen in einem Durchlauf eine PDF und eine DXF Datei.

**Hinweise**

delegate[m] ist nur wirksam, wenn converter[n].type = multiple gesetzt ist.

Der Index m in delegate[m] muss mit 0 beginnen und fortlaufend um 1 erhöht werden. (delegate[0], delegate[1], delegate[2] etc.)

Der Wert key von delegate[m] = key muss einem konfigurierten Index converter[key] entsprechen.

**8.11 converter[n].hide[m].part**

Mit diesem Schlüssel können die Namen von Teilen definiert werden, die von COP Dok nicht ausgegeben werden. COP Dok blendet diese Teile aus, so dass die Teile und dessen Inhalt in der Ausgabedatei nicht sichtbar sind.

**Beispiel**

```
converter[pdf].hide[0].part = .confidential
converter[pdf].hide[1].part = .delete
```

**8.12 converter[n].hide[m].info**

Mit diesem Schlüssel können Infotexte definiert werden. COP Dok blendet alle Geometrieelemente aus, die diesen Infotext haben, so dass die Geometrieelemente in der Ausgabedatei nicht sichtbar sind.

**Beispiel**

```
converter[pdf].hide[0].info = MARKED_AS_CONFIDENTIAL
```

**8.13 converter[n].pdf.merge**

Bei der PDF Ausgabe einer Zeichnung mit mehreren Blättern werden die Blätter in einzelne PDF Dateien exportiert. Falls der Wert dieses Schlüssels gleich 1 ist, werden alle Blätter in einer einzigen PDF Datei zusammengefügt.

Damit COP Dok Zeichnungen in eine PDF Datei zusammenfügen kann, muss die Software Ghostscript auf dem System installiert sein (siehe Installation). Der Ordner in dem Ghostscript installiert ist, muss im Schlüssel converter.gs.dir definiert sein.

**Standardwert**

0

**Beispiel**

converter[pdf].pdf.merge = 1

**8.14 converter[n].scale**

Dieser Schlüssel definiert eine Skalierung. Die Zeichnung wird vor der Ausgabe mit diesem Wert skaliert.

**Standardwert**

1

**Beispiel**

converter[pdf].scale = 0.989

**8.15 converter[n].hiddenline.enable**

Falls die Zeichnung sog. Hidden Line Elemente enthält, bereitet COP Dok vor der Ausgabe der Zeichnung mit dem Drafting Befehl HL\_GENERATE\_HIDDEN die versteckten Kanten auf. Mit diesem Schlüssel kann die Aufbereitung der versteckten Kanten ein- oder ausgeschaltet werden. Dieser Schlüssel überschreibt den Wert des Schlüssels hiddenline.enable.

**Gültige Werte**

1 = versteckte Kanten aufbereiten

0 = versteckte Kanten nicht aufbereiten

**Standardwert**

1

**Beispiel**

converter[pdf].hiddenline.enable = 0

**Siehe auch**

hiddenline.enable

## 8.16 **converter[n].post.cmd**

Diese Schlüssel definiert ein Programm, welches von COP Dok nach Abschluss der Aufgabe gestartet wird. COP Dok übergibt dem Programm den Namen (Pfad) der Ausgabedatei. Mit <outputfile> kann die Position des Ausgabedateinamen innerhalb des Befehls definiert werden.

### **Standardwert**

keiner

### **Beispiele**

converter[0].post.cmd= myprogramm.exe

COP Dok startet das Programm (Skript) myprogramm.exe und übergibt den Ausgabedateinamen.

converter[0].post.cmd= my\_batch.bat -f <outputfile> -print

COP Dok startet das Programm (Skript) myprogramm.exe und übergibt den Ausgabedateinamen an der entsprechenden Position (nach -f).

converter[0].post.cmd= <outputfile>

COP Dok führt Ausgabedatei aus, als ob es ein Programm wäre. In diesem Fall startet Windows die mit der Datei verknüpfte Applikation (zum Beispiel bei einer PDF Datei den Acrobat Reader).

## 9 Grafikdateien Ausgabe

COP Dok kann Grafikdateien, wie JPEG, GIF, TIFF etc. ausgeben. Hierfür muss das Programm Ghostscript installiert sein.

### 9.1 **converter.gs.dir**

Dieser Schlüssel definiert das Installationsverzeichnis von Ghostscript.

**Standardwert**

-

**Beispiel**

converter.gs.dir = C:\Program Files (x86)\gs\gs8.71

### 9.2 **converter[n].density**

Dieser Schlüssel definiert die Auflösung der Grafik-Ausgabedatei.

**Standardwert**

75

**Beispiel**

converter[0].density= 150

**Hinweis**

Dieser Schlüssel wird nur bei der Ausgabe einer Grafikdatei (JPEG, TIFF etc.) berücksichtigt. Grosse Werte können sich negativ auf die Systemauslastung (Speicher, Performance) auswirken.

### 9.3 **converter[n].geometry**

Dieser Schlüssel definiert die Grösse der Grafikdatei.

**Standardwert**

-

**Beispiel**

converter[tiff].geometry=768x768

### 9.4 **converter[n].converter.options**

Dieser Schlüssel definiert Optionen (Argumente), die an den Grafik Konverter übergeben werden. (siehe <http://www.imagemagick.org>, Command Line Tool convert)

**Standardwert**

-

**Beispiel**

converter[tiff] .converter.options = -depth 8

## 10 PKM Konfigurationen

COP Dok kann PKM Daten laden und verarbeiten. PKM Dateien werden von Creo Elements/Direct Manager (Model Manager oder Drawing Manager) exportiert. In PKM Dateien sind nebst der Zeichnung noch weitere Daten (Zeichnungsrahmen, Stammdaten, Zeichnungsattribute etc.) enthalten.

### 10.1 pkm.changenotes.max

Dieser Schlüssel definiert, wie viele Änderungsnotizen im Zeichnungskopf aufgedruckt werden. Es werden jeweils die höchsten Indexe aufgedruckt. Der Wert 0 bedeutet, dass alle Änderungsnotizen aufgedruckt werden.

#### Standardwert

0

#### Beispiel

pkm.changenotes.max = 3

Wenn die Änderungsnotizen mit Index A, B, C, D und E vorhanden sind, dann werden in diesem Beispiel nur die drei Änderungsnotizen C, D und E aufgedruckt.

### 10.2 pkm.lang

Dieser Schlüssel definiert die Sprache des Datenbankschemas.

Wert	Sprache
en	Englisch
de	Deutsch

#### Standardwert

en

#### Beispiel

pkm.lang = de

### 10.3 pkm.reload.frame

Dieser Schlüssel definiert, ob COP Dok den Zeichnungsrahmen aus der PKM Datei in die Zeichnung nachladen soll.

#### Standardwert

1

#### Beispiel

pkm.reload.frame = 1

### 10.4 pkm.frame.determines.format

COP Dok bestimmt das Ausgabeformat (A0, A1 etc.) anhand der Grösse der Zeichnung, resp. des Blattes. COP Dok berechnet hierfür die Box um die gesamte Geometrie. Falls der Wert des Schlüssels pkm.frame.determines.format gleich 1 ist, bestimmt COP Dok das Ausgabeformat anhand der Grösse des Zeichnungsrahmens. Sämtliche Geometrie, die ausserhalb des Rahmens liegt, wird nicht ausgegeben oder abgeschnitten.

#### Standardwert

0

#### Beispiel

pkm.frame.determines.format = 1

#### Siehe auch

pkm.tb.prefix[n]

## 10.5 **pkm.tb.prefix[n]**

Der Zeichnungsrahmen ist in der Zeichnung (oder Blatt) in der Regel in einem separaten Teil erfasst. Der Teilname setzt sich aus einer Präfix und dem Format des Zeichnungsrahmens zusammen (Beispiel für A0 Rahmen: .tb\_frame\_A0).

Der Schlüssel pkm.tb.prefix[n] (n = 0,1,2 ...) definiert die möglichen Präfixe für den Teilnamen der Zeichnungsrahmen. Die Standard Präfixe sind in der Konfigurationsdatei default.ini definiert.

### **Standardwert**

siehe default.ini

### **Beispiel**

```
pkm.tb.prefix[0]=.tb_frame_  
pkm.tb.prefix[1]=.sfeld_rahmen_  
pkm.tb.prefix[2]=.cadre_ct_
```

## 10.6 **pkm.tb.log**

COP Dok schreibt während der Ausführung diverse Informationen in Dateien im Ordner %TEMP%\lpmilog. U.a. werden auch alle Daten (Stammdaten, Dokumentdaten etc.) aus der PKM Dateien in die LOG Dateien geschrieben. Dies kann die Ausführungsdauer merklich verlängern. Falls pkm.tb.log= 0 ist, schreibt COP Dok die Daten nicht in die LOG Dateien.

### **Standardwert**

1

### **Beispiel**

```
pkm.tb.log= 0
```

## 10.7 **pkm.tb.update.macro.prev**

COP Dok aktualisiert beim Laden einer PKM Datei die Zeichnungsrahmen mit den Daten (Stammdaten, Dokumentdaten etc.) aus der PKM Datei. Mit dem Schlüssel pkm.tb.update.macro.prev kann ein Makro definiert werden, das von COP Dok vor der Aktualisierung des Zeichnungsrahmens ausgeführt wird.

### **Standardwert**

-

### **Beispiel**

```
pkm.tb.update.macro.prev = my_macro_before_tb_update
```

## 11 defaults.m

In der Datei defaults.m sind Makros definiert, die COP Dok lädt und vor oder nach dem Druck-, resp. Konverter- Prozess ausführt (siehe entsprechende Konfigurationen in lpmi.ini). Diese Datei sollte nicht geändert werden, da diese während einer Neuinstallation von COP Dok oder während der Installation eines Updates vom Installationsprogramm überschrieben wird.

Der Inhalt dieser Datei kann als Vorlage für eine eigene Makrodatei (zum Beispiel customize.m) verwendet werden. Mit dem Schlüssel me.customize.file kann die eigene Makrodatei registriert werden. COP Dok liest die Datei während dem Start ein. Der Inhalt von defaults.m ist in den folgenden Kapiteln dokumentiert.

### 11.1 lpmi\_c\_plot\_set\_plot\_transformation\_generic

Dieses Makro definiert die Standard Plot Transformation (Stifte, Strichstärken etc.). Bei einer Standardinstallation wird dieses Makro unmittelbar vor dem Druck- oder Konvertierungsprozess ausgeführt.

Mit den Schlüsseln printer[n].macro.transformation und converter[n].macro.transformation können eigene Transformationsmakros registriert werden.

### 11.2 lpmi\_c\_plot\_set\_plot\_transformation\_a4 (\_a3, \_a2 etc.)

Diese Makros definieren die Standard Plot Transformation (Stifte, Strichstärken etc.) für spezifische Zeichnungsformate A0 bis A4. Bei einer Standardinstallation wird dieses Makro unmittelbar vor dem Druck- oder Konvertierungsprozess ausgeführt.

Mit den Schlüsseln printer[n].macro.transformation.a[0..4] und converter[n].macro.transformation.a[0..4] können eigene Transformationsmakros registriert werden.

### 11.3 lpmi\_plot\_g\_line\_scale

Diese globale Makrovariable wird vom System berechnet. Die Variable enthält einen Skalierungswert, falls die Zeichnung von einem Format auf ein anderes Format skaliert wird, zum Beispiel wenn eine A3 Zeichnung auf ein A4 Papier ausgegeben wird.

Diese Variable kann verwendet werden, um die Strichstärken an das Ausgabeformat anzupassen.

Beispiel

```
DEFINE lpmi_c_plot_set_plot_transformation_generic
  TRUE_COLOR_PLOTTING OFF
  PLOT_TRANSFORMATION RESET
  PLOT_TRANSFORMATION ALL 0 1 0 1 0 1 SAME 1
  PLOT_TRANSFORMATION ALL 0.5 1 0 0.5 0 0.5 0 0 SAME PENWIDTH (0.4 * lpmi_plot_g_line_scale) 1
  PLOT_TRANSFORMATION ALL 0 0.5 0.5 1 0 0.5 0 0 SAME PENWIDTH (0.3 * lpmi_plot_g_line_scale) 1
END_DEFINE
```

### 11.4 lpmi\_plot\_delete\_elem

Dies ist ein Beispielmakro für den Schlüssel printer.macro.load.post.

## 12 DXFDWG.con

Die DXF/DWG Konversion erfolgt mit dem Creo Elements/Direct Drafting Konverter dxfdwg.exe. Die Konfiguration des Konverters ist in der Standarddatei DXFDWG.con definiert. Angaben zu den Einstellungen und Möglichkeiten des Konverters können aus der Creo Elements/Direct Drafting Dokumentation entnommen werden.

### 12.1 dxfdwg.config.file

Ist der Schlüssel dxfdwg.config.file definiert, dann kopiert COP Dok die DXFDWG Datei in den Installationsordner des DXFDWG Konverters. Dies bedingt, dass die Benutzer die entsprechenden Schreibrechte auf den Zielordner haben.

dxfdwg.config.file = C:\data\dev\copdok\DXFDWG.con

#### **Standardwert**

-

#### **Beispiel**

dxfdwg.config.file = C:\data\dev\copdok\DXFDWG.con

## 13 Fehlersuche (Troubleshooting)

Falls COP Dok nicht die gewünschten Resultate liefert, können für die Fehlersuche diverse Konfigurationen definiert werden.

COP Dok schreibt umfangreiche LOG Informationen in verschiedene Dateien in %TEMP%\lpmilog. Für jede Aufgabe erstellt COP Dok eine separate LOG Datei.

### 13.1 me.transmode

COP Dok führt Creo Elements/Direct Drafting, 2D Access oder Modeling im Hintergrund aus. Falls die Variable me.transmode = 0 ist, führt COP Dok Creo Elements/Direct Drafting, 2D Access oder Modeling nicht im Hintergrund aus.

#### Standardwert

1

#### Beispiel

me.transmode = 0

### 13.2 debug.exit

COP Dok beendet Creo Elements/Direct Drafting, 2D Access oder Modeling automatisch, nach dem es die Aufgabe ausgeführt hat. Falls debug.exit = 0 ist, beendet COP Dok die Programme nicht.

#### Standardwert

1

#### Beispiel

debug.exit = 0

### 13.3 debug.process.start

COP Dok startet Creo Elements/Direct Drafting oder 2D Access und führt die Aufgabe sofort aus. Falls debug.process.start = 0 ist, startet COP Dok die Programme, führt aber die Aufgabe nicht aus. Die Aufgabe kann durch Eingabe des Befehls "lpmi\_start" in der Befehlszeile von Creo Elements/Direct Drafting oder 2D Access gestartet werden.

#### Standardwert

1

#### Beispiel

debug.process.start = 0

## 14 Konfigurationsbeispiele

Ein ausführliches Beispiel einer COP Dok Konfigurationsdatei finden Sie im Installationsverzeichnis unter `.\newconfig\beispiel.ini`.

Beispiele für Plot Transformation Makros finden Sie im Installationsverzeichnis unter `.\defaults.m`