

LineMetrics Offline Sync Tool - Data Monk

```
Version:          1.6
Datum:           10.05.2017
Verantwortlicher: Thomas Pillmayr <t.pillmayr@linemetrics.com>
```

Data Monk ist ein Tool von LineMetrics, dass OpenSource zur Verfügung gestellt wird und es Kunden erlaubt, Daten aus der LineMetrics Cloud in ein kundenspezifisches lokales Format zu synchronisieren.

Versionsänderung von 1.5.x auf 1.6

Wichtig: Da die neue Version (1.6) nun auch die neuere LM3 Api unterstützt muss die Konfiguration hinsichtlich der Datastreams geändert werden. Anstatt wie bisher (<1.6) muss auch der Datastream Eintrag in der Konfiguration mit einem Key versehen werden, ähnlich wie es bei den Jobs ist. Diese Änderung bezieht sich auch auf die Meta-Infos.

Version < 1.6

```
job.1.datastream=8782
meta.datastream.8782.customer_id=1234
```

Version 1.6+

```
job.1.datastream.1.id=8782
meta.datastream.1.customer_id=1234
```

Funktionalität

- Frei definierbare zeitliche Synchronisierung (ähnlich Cronjobs)
- Hinterlegen von Meta Attributen zur Erweiterung/Anreicherung der Daten
- Benutzerdefinierte Komprimierung von Daten (Beispielweise: Summe/Durchschnitt pro 15 Minuten)
- Aktuell unterstützte Speicherformate: CSV

Anforderungen

- Java JRE 1.7+
- Internetverbindung
- gültiges LineMetrics Konto mit konfigurierter sicherer API im [LineMetrics Cloud Device Manager](#)

TODO

- Abfangen von Fehlern, wenn Sync Vorgang Fehler aufweist
- JDBC Storage Plugin
- grafische Oberfläche zur Konfiguration der Synchronisierungs Vorgänge

Erste Schritte

Es ist vorgesehen in einer späteren Version eine grafische Oberfläche zur Konfiguration der Synchronisierungsvorgänge anzubieten. Aktuell muss die Konfiguration in einem Konfigurationsfile `system.properties` erfolgen.

Aufbau `system.properties`

1. API Zugangsdaten
2. Synchronisierungseinstellungen
3. Aktivierte Synchronisierungsvorgänge
4. Meta Informationen

API Zugangsdaten

```
#Version 1 API
api.endpoint=http://bapi.linemetrics.com:8002
api.hash=ABCDEFGF...

#Version 2 API
api.client_id=ABCD
api.client_secret=ABCD

#API version
api.version=com.linemetrics.monk.apiv2
```

`api.endpoint` definiert den API Server Endpunkt und sollte eigentlich immer gleich sein.

`api.hash` ist der eigentliche API Zugang und verifiziert den Zugriff über die API. Den Hash findest du im [LineMetrics Cloud Device Manager](#), wenn du auf deine konfigurierte API Schnittstelle und dann auf den Button "Zugangsdaten" klickst.

Wird Version 2 der API verwendet muss anstatt `api.hash` der Wert für `api.client_id` und `api.client_secret` angegeben werden. `api.version` gibt an welche Version der API verwendet werden soll. Mögliche Werte sind `com.linemetrics.monk.apiv2` und `com.linemetrics.monk.api.`

Synchronisierungseinstellungen

Die Synchronisierungseinstellungen unterteilen sich in 4 Bereiche:

- Schedulingeinstellungen / Wie oft soll synchronisiert werden?

- Welche Daten sollen synchronisiert werden?
- Sollen die Daten vor dem Speichern verarbeitet werden?
- Wie sollen die Daten gespeichert werden?

Schedulereinstellungen

```
# Einstellungen haben immer folgendes Format
# job.[JOB ID].info.[EINSTELLUNG]=[WERT]

job.1.info.scheduler_mask=0 0 8-17 ? * MON-SAT
job.1.info.timezone=Europe/Vienna
job.1.info.batch_size=PT1m
job.1.info.duration=PT1H
```

Alle Einstellungen die zu einem Sync Vorgang gehören, müssen die selbe `JOB ID` beinhalten. Die `JOB ID` ist ein ganzzahliger numerischer Wert, der selbst definiert werden kann und lediglich in der Konfiguration konsistent sein muss.

Mit der Einstellung `scheduler_mask` wird konfiguriert, wie oft die Daten synchronisiert werden sollen.

In der oben angeführten Einstellung wird der Synchronisierungsvorgang Montag-Samstag zwischen 8:00 und 17:00 zu jeder vollen Stunde ausgeführt. Weitere mögliche Einstellungen findest du [hier](#).

Zum Testen gibt es noch die Möglichkeit bei der `scheduler_mask` den Wert `NOW` zu hinterlegen, was dazu führt, dass der Vorgang nur einmal beim Starten des Programmes ausgeführt wird. Diese Funktionalität dient vorwiegend zum Testen einer neuen Konfiguration, ohne dass man jedesmal auf einen bestimmten zeitlichen Trigger warten muss.

`timezone` regelt das Zeitformat der gespeicherten Daten. In der LineMetrics Cloud werden alle Daten im UTC Format gespeichert. Daten werden erst zur Laufzeit (zum Abfragezeitpunkt) in das für den Benutzer passende Format umgewandelt.

`batch_size` legt fest, mit welcher Granularität die Daten aus der API geladen werden sollen. Angaben im ISO_8601 Standard. Mögliche Werte: `PT1M`, `PT1H`, `P1D`

`duration` legt fest, was für ein Zeitraum geladen werden soll. Angaben im ISO_8601.

Aus den Einstellungen lässt sich interpretieren, dass Montag-Samstag zwischen 8:00 und 17:00 Uhr zu jeder vollen Stunde die vergangene Stunde als Minuten Datenpunkte abgeholt wird.

Welche Daten sollen synchronisiert werden?

```
job.1.datastream.1.id=123
job.1.datastream.2.id=456
```

Pro Zeile kann ein Datenstrom zum Sync Vorgang hinzugefügt werden. Die für das Konto

verfügbaren Datenströme können über die [Datenstrom Übersichtsseite](#) eruiert werden.

Daten-Verarbeitung

Möglicherweise sind die von der LineMetrics Cloud geladenen Daten noch im falschen Format. Über die Datenverarbeitung besteht die Möglichkeit, diese Daten noch zusätzlich zu verarbeiten/aggregieren.

In der aktuellen Version gibt es ein Plugin zum Aggregieren der Daten von der Minuten Granularität in ein 15 Minuten Datenpaket.

```
job.1.processor.type=com.linmetrics.monk.processor.plugins.compress.CompressorPlugin
job.1.processor.compression_mode=SUM
job.1.processor.compression_size=PT15M
```

Mit dem Attribut `compression_mode` wird festgelegt, ob die Summe `SUM` oder der Durchschnitt `AVG` gebildet werden soll.

Mit dem Attribut `compression_size` kann festgelegt werden, wieviele Daten zusammengefasst werden sollen. Angaben wieder im Format ISO_8601. In unserem Fall summieren wir die 1 Minuten-Daten-Pakete von der API zu jeweils 15 Minuten-Pakete.

Daten-Speicherung

Am Ende des Sync Vorganges sollen die Daten persistiert werden. Dies passiert im folgenden Abschnitt

```
job.1.store.1.type=com.linmetrics.monk.store.plugins.csv.StorePlugin
job.1.store.1.csv_number_locale=de_AT
job.1.store.1.csv_file_path=exports/
job.1.store.1.csv_file_template=${job.start:YYYY-mm-dd}.csv
job.1.store.1.csv_header_template=Das ist der Header meiner CSV
job.1.store.1.csv_line_template=${item.start:YYYY-mm-dd
HH:mm:ss};${item.end:YYYY-mm-dd HH:mm:ss};${item.value:0.00}
```

Die Konfiguration folgt dabei folgenden Aufbau:

```
job.[JOB ID].store.[STORE ID].[ATTRIBUTE]=[WERT]
```

Die Store ID kann beliebig vergeben werden und ermöglicht das verwenden von mehreren Persistierungsarten innerhalb eines Jobs. In diesem Beispiel wird als Store ID eine ansteigende numerische Reihenfolge verwendet, wobei Store 1 die Daten in ein CSV persistiert und Daten bei einer bestehenden Datei anhängt werden. Im nächsten Abschnitt wird die Persistierung über das Prefilled Plugin vorgestellt, welche als zweiter Store mit der ID 2 konfiguriert wird.

Das Attribut `csv_number_locale` legt das Locale für den Export Vorgang fest. Dies ist vor allem für die richtige Formatierung von numerischen Werten wichtig.

Das Attribut `csv_file_path` legt den Speicherort (Ordner) der CSV Datei fest.

Das Attribut `csv_file_template` legt den Dateinamen der CSV Datei fest. **Wichtig:** Bei allen Template Attributen können Platzhalter verwendet werden, die erst während des Sync Vorgangs mit Werten befüllt werden. Platzhalter werden im einem späteren Abschnitt genauer behandelt.

Das Attribut `csv_header_template` legt den Kopf (die ersten Zeilen) der CSV Datei fest, wenn die Datei neu angelegt wird. Falls die Datei, in der die Daten geschrieben werden sollen, bereits existiert, werden die Daten (ohne HEADER) lediglich angehängt.

Das Attribut `csv_line_template` legt das Format fest, wie die einzelnen Datenpunkte in das CSV File geschrieben werden sollen.

Daten-Speicherung mit Prefilled Plugin

Das Prefilled Plugin ermöglicht die Persistierung der Daten ähnlich dem CSV Plugin, allerdings mit dem Unterschied, dass die CSV Datei beim Anlegen bereits mit allen konfigurierten Zeitstempeln gefüllt wird. Die Daten werden dann erst Schritt für Schritt in die Datei eingefügt.

Beispiel Konfiguration für das Prefilled Plugin

```
job.1.store.2.type=com.linometrics.monk.store.plugins.csv.PrefilledPlugin
job.1.store.2.csv_time_scope=PTD
job.1.store.2.csv_time_slice=PT15M
job.1.store.2.csv_number_locale=de_AT
job.1.store.2.csv_file_path=exports/
job.1.store.2.csv_file_template=${job.start:YYYY-mm-dd}.csv
job.1.store.2.csv_header_template=Das ist der Header meiner CSV
job.1.store.2.csv_empty_line_template=${item.start:YYYY-mm-dd
HH:mm:ss};${item.end:YYYY-mm-dd HH:mm:ss};
job.1.store.2.csv_line_template=${item.start:YYYY-mm-dd
HH:mm:ss};${item.end:YYYY-mm-dd HH:mm:ss};${item.value:0.00}
```

Neben den Attributen des CSV Plugin sind noch folgende weitere Attribute zu konfigurieren:

Das Attribut `csv_time_scope` legt fest, über welchen Zeitraum sich eine einzelne Datei erstreckt. Diese Angabe ist notwendig, da das System nicht automatisch über den Dateinamen den Zeitraum extrahieren kann. Angaben im ISO_8601 Format. P1D = 1 Tag

Das Attribut `csv_time_slice` legt fest, über welchen Zeitraum sich ein einzelner Datenpunkt erstreckt. Angaben im ISO_8601 Format. PT15M = 15 Minute

Das Attribut `csv_empty_line_template` legt das Format fest, wie die leeren Datenpunkte, die beim Anlegen einer neuen Datei für den gesamten Zeitraum geschrieben werden, aussehen sollen. Diese Zeilen werden später durch die tatsächlichen Werte bzw. durch die Konfiguration von

csv_line_template ersetzt.

Daten-Weiterleitung an Mail-Adresse

Das Mail Plugin ermöglicht das Versenden der Daten als CSV Datei an eine oder mehrere Email-Adressen.

Beispiel Konfiguration für das Mail Plugin:

```
job.1.store.1.type=com.linmetrics.monk.store.plugins.mail.MailPlugin
job.1.store.1.csv_number_locale=de_AT
job.1.store.1.csv_file_template=${job.start:yyyy-MM-dd}.csv
job.1.store.1.csv_header_template=Das ist der Header meiner CSV mit einer Meta Info
${meta.clientID}
job.1.store.1.csv_line_template=${item.start:yyyy-MM-dd
HH:mm:ss};${item.end:yyyy-MM-dd HH:mm:ss};${item.value:0.00};${meta.customer_id}
job.1.store.1.csv_line_separator=<LF>
job.1.store.1.mail_receiver=empfaenger@test.com
job.1.store.1.mail_subject=Linmetrics CSV Export
job.1.store.1.mail_attachment=true
job.1.store.1.mail_smtp_host=smtp.test.com
job.1.store.1.mail_smtp_sender=sender@test.com
job.1.store.1.mail_smtp_user=username
job.1.store.1.mail_smtp_password=password
job.1.store.1.mail_smtp_port=465
job.1.store.1.mail_smtp_tlseabled=true
```

Neben den Attributen des CSV Plugin sind noch folgende weitere Attribute zu konfigurieren:

Das Attribut `mail_receiver` legt fest an welche Email-Adressen die Daten gesendet werden sollen. Mit Beistrich getrennt können auch mehrere Email-Adressen angegeben werden.

Das Attribut `mail_subject` legt den Betreff fest.

Das Attribut `mail_attachment` kann die Werte `true` und `false` besitzen und entscheidet darüber ob die Daten als CSV-Datei im Anhang versendet werden sollen (=true) oder direkt als Text in die Email geschrieben werden sollen.

Das Attribut `mail_smtp_host` gibt den SMTP-Host an, über den die Email verschickt werden soll. `mail_smtp_sender` gibt die Sender-Email Adresse an. `mail_smtp_user` und `mail_smtp_password` konfigurieren den User der am SMTP-Host über die nötigen Berechtigungen verfügt. Zusätzlich muss mit `mail_smtp_port` noch der Port des SMTP-Host angegeben werden und ob der SMTP-Host TLS verwenden soll (`mail_smtp_tlseabled`).

Daten-Weiterleitung an LineMetrics v3

Das Bridge Plugin ermöglicht das Weiterleiten von Daten aus der v2 Umgebung in die v3 Umgebung. Wichtig hier ist, dass zuvor in der v3 Umgebung ein gültiger REST Api Zugang und das Objektmodell (inklusive Messpunkte mit eindeutiger Adressierung über Custom-Key & Alias ->

Details [hier](#)) angelegt wurden.

Beispiel Konfiguration für das Bridge Plugin

```
job.1.store.1.type=com.linometrics.monk.store.plugins.lm3.BridgePlugin
job.1.store.1.oauth_client_id=api_5a...
job.1.store.1.oauth_client_secret=0a59...
job.1.store.1.connection_url=https://lm3api.linometrics.com/v2/data/
job.1.store.1.oauth_client_url=https://lm3api.linometrics.com/oauth/access_token
job.1.store.1.items_per_request=1024
```

Über die Attribute `oauth_client_id` und `oauth_client_secret` werden die Credentials für den REST Api Zugang der v3 Umgebung definiert.

Das Attribut `connection_url` definiert den Endpunkt zum Schreiben der eigentlichen Nutzdaten und das Attribut `oauth_client_url` definiert den Endpunkt zum Erzeugen des OAuth Authentifizierungstoken. Diese Einstellungen müssen im Normalfall nicht adaptiert werden.

Über das Attribut `items_per_request` kann die Größe der Batches definiert werden, welche über die REST Api innerhalb eines Requests übertragen werden.

Aktivierte Synchronisierungsvorgänge

Es müssten nicht immer zwingend alle Synchronisierungsvorgänge gestartet werden. Durch das Attribut `activated_jobs` kann genau festgelegt werden, welcher Vorgang berücksichtigt und welcher ignoriert werden soll. **Wichtig:** Pro aktiviertem Vorgang eine eigene Zeile. Das Attribut wird nicht überschrieben.

Meta Informationen

Oft macht es Sinn die exportierten Daten mit zusätzlichen semantischen Daten zu versehen, um bspw. Daten später besser mit anderen Stammdaten verknüpfen zu können.

```
#Meta Informationen haben immer folgenden Aufbau
meta.(job oder datastream).[ID].[KEY]=[VALUE]

meta.job.1.data_type=Energy Consumption
meta.datastream.1.customer_id=1234
```

Hier wird zuerst eine Meta Information für den Synchronisierungsvorgang mit der ID 1 angelegt und dann eine Info für den Datenstrom mit dem Key 1 (bezieht sich auf `job.1.datastream.1.id=123`). Meta Informationen können bei den Templates in Form von Platzhaltern verwendet werden. Platzhalter werden im nächsten Abschnitt behandelt.

Platzhalter

Arten von Platzhalter

Art	**Beschreibung**	**Formatierung**
Meta	Meta Informationen die über Konfiguration spezifiziert worden sind	NEIN
Job	Daten die im Context des Synchronisationsvorgang stehen	JA
Item	Daten im Context eines einzelnen Datenpunkts	NEIN

Variablen

Art	Variable	Beschreibung	Formattierung
Meta	<code>\${meta.[KEY]}</code>	Gibt je nach Context den Wert des Meta Info Schlüssels zurück. Wenn der Schlüssel nicht existiert wird UNDEFINED ausgegeben.	
Job	<code>\${job.start.[FORMAT]}</code>	Sync Zeitbereich Startzeitpunkt	Formatierung Zeit
	<code>\${job.end.[FORMAT]}</code>	Sync Zeitbereich Endzeitpunkt	Formatierung Zeit
	<code>\${job.timezone.[FORMAT]}</code>	Zeitzone	
Item	<code>\${item.start.[FORMAT]}</code>	Datenpunkt Zeitbereich Start	Formatierung Zeit
	<code>\${item.end.[FORMAT]}</code>	Datenpunkt Zeitbereich Ende	Formatierung Zeit
	<code>\${item.min.[FORMAT]}</code>	Datenpunkt Min Wert (falls verfügbar)	Formatierung Zahl
	<code>\${item.max.[FORMAT]}</code>	Datenpunkt Max Wert (falls verfügbar)	Formatierung Zahl
	<code>\${item.value.[FORMAT]}</code>	Datenpunkt Wert	Formatierung Zahl

Formatierungen

Zeit

Definition laut ISO8601:

```
YYYY                (z.B. 1997)
YYYY-MM            (z.B. 1997-07)
YYYY-MM-DD         (z.B. 1997-07-16)
YYYY-MM-DDThh:mmTZD (z.B. 1997-07-16T19:20+01:00)
YYYY-MM-DDThh:mm:ssTZD (z.B. 1997-07-16T19:20:30+01:00)
```

Zahl

```
#,##              12,675    =>    12,67
00000,00          12,6      =>    00012,60
0.000,##         1212,6     =>    1.212,6
0.000,00         1212,6     =>    1.212,60
#                2,3        =>    2
```

Vollständige Konfigurationen

Periodisches Exportieren in Dateien

```
api.endpoint=http://bapi.linemetrics.com:8002
```



```
api.hash=ABCDEFGF...
api.version=com.linemetrics.monk.api

job.1.info.scheduler_mask=0 0 8-17 ? * MON-SAT
job.1.info.timezone=Europe/Vienna
job.1.info.batch_size=PT1m
job.1.info.duration=PT1H

job.1.datastream.1.id=123
job.1.datastream.2.id=456

job.1.processor.type=com.linemetrics.monk.processor.plugins.compress.CompressorPlug
in
job.1.processor.compression_mode=SUM
job.1.processor.compression_size=PT15M

job.1.store.1.type=com.linemetrics.monk.store.plugins.csv.StorePlugin
job.1.store.1.csv_number_locale=de_AT
job.1.store.1.csv_file_path=exports/
job.1.store.1.csv_file_template=${job.start:YYYY-mm-dd}.csv
job.1.store.1.csv_header_template=Das ist der Header meiner CSV
job.1.store.1.csv_line_template=${item.start:YYYY-mm-dd
HH:mm:ss};${item.end:YYYY-mm-dd HH:mm:ss};${item.value:0.00}

job.1.store.2.type=com.linemetrics.monk.store.plugins.csv.PrefilledPlugin
job.1.store.2.csv_time_scope=P1D
job.1.store.2.csv_time_slice=PT15M
job.1.store.2.csv_number_locale=de_AT
job.1.store.2.csv_file_path=exports/
job.1.store.2.csv_file_template=${job.start:YYYY-mm-dd}.csv
job.1.store.2.csv_header_template=Das ist der Header meiner CSV
job.1.store.2.csv_empty_line_template=${item.start:YYYY-mm-dd
HH:mm:ss};${item.end:YYYY-mm-dd HH:mm:ss};
job.1.store.2.csv_line_template=${item.start:YYYY-mm-dd
HH:mm:ss};${item.end:YYYY-mm-dd HH:mm:ss};${item.value:0.00}

activated_jobs=1

meta.job.1.data_type=Energy Consumption
meta.datastream.1.customer_id=1234
```

Periodisches Übertragen an REST Endpunkt

```
api.endpoint=http://bapi.linemetrics.com:8002
api.hash=ABCDEFGF...
api.version=com.linemetrics.monk.api

job.1.info.scheduler_mask=0 0 8-17 ? * MON-SAT
job.1.info.timezone=Europe/Vienna
job.1.info.batch_size=PT1m
job.1.info.duration=PT1H

job.1.datastream.1.id=123
job.1.datastream.2.id=456

job.1.store.1.type=com.linemetrics.monk.store.plugins.lm3.BridgePlugin
```

```
job.1.store.1.connection_url=https://lm3api.linemetrics.com/v2/data/
job.1.store.1.oauth_client_id=api_ABC...
job.1.store.1.oauth_client_secret=ABC...
job.1.store.1.oauth_client_url=https://lm3api.linemetrics.com/oauth/access_token
job.1.store.1.items_per_request=1024

activated_jobs=1
meta.datastream.1.custom_key=Maschine
meta.datastream.2.alias=Messpunkt1

meta.datastream.1.custom_key=Maschine
meta.datastream.2.alias=Messpunkt2
```

Periodisches Versenden an Mail-Adresse

```
api.endpoint=http://bapi.linemetrics.com:8002
api.hash=ABCDEFG...
api.version=com.linemetrics.monk.api

job.1.info.scheduler_mask=0 0 8-17 ? * MON-SAT
job.1.info.timezone=Europe/Vienna
job.1.info.batch_size=PT1m
job.1.info.duration=PT1H

job.1.datastream.1.id=123
job.1.datastream.2.id=456

job.1.store.1.type=com.linemetrics.monk.store.plugins.mail.MailPlugin
job.1.store.1.csv_number_locale=de_AT
job.1.store.1.csv_file_template=${job.start:yyyy-MM-dd}.csv
job.1.store.1.csv_header_template=Das ist der Header meiner CSV mit einer Meta Info
${meta.clientID}
job.1.store.1.csv_line_template=${item.start:yyyy-MM-dd
HH:mm:ss};${item.end:yyyy-MM-dd HH:mm:ss};${item.value:0.00};${meta.customer_id}
job.1.store.1.csv_line_separator=<LF>
job.1.store.1.mail_receiver=empfaenger@test.com
job.1.store.1.mail_subject=Linemetrics CSV Export
job.1.store.1.mail_attachment=true
job.1.store.1.mail_smtp_host=smtp.test.com
job.1.store.1.mail_smtp_sender=sender@test.com
job.1.store.1.mail_smtp_user=username
job.1.store.1.mail_smtp_password=password
job.1.store.1.mail_smtp_port=465
job.1.store.1.mail_smtp_tlsenabled=true

activated_jobs=1
meta.datastream.1.custom_key=Maschine
meta.datastream.2.alias=Messpunkt1

meta.datastream.1.custom_key=Maschine
meta.datastream.2.alias=Messpunkt2
```

Periodisches Exportieren in Dateien unter Verwendung der APIv2

```

api.client_id=ABCD...
api.client_secret=ABCDEFG...
api.version=com.linemetrics.monk.apiv2

job.1.info.scheduler_mask=0 0 8-17 ? * MON-SAT
job.1.info.timezone=Europe/Vienna
job.1.info.batch_size=PT1m
job.1.info.duration=PT1H

job.1.datastream.1.customkey=123
job.1.datastream.1.alias=123

job.1.datastream.2.customkey=123
job.1.datastream.2.alias=123

job.1.processor.type=com.linemetrics.monk.processor.plugins.compress.CompressorPlug
in
job.1.processor.compression_mode=SUM
job.1.processor.compression_size=PT15M

job.1.store.1.type=com.linemetrics.monk.store.plugins.csv.StorePlugin
job.1.store.1.csv_number_locale=de_AT
job.1.store.1.csv_file_path=exports/
job.1.store.1.csv_file_template=${job.start:YYYY-mm-dd}.csv
job.1.store.1.csv_header_template=Das ist der Header meiner CSV
job.1.store.1.csv_line_template=${item.start:YYYY-mm-dd
HH:mm:ss};${item.end:YYYY-mm-dd HH:mm:ss};${item.value:0.00}

job.1.store.2.type=com.linemetrics.monk.store.plugins.csv.PrefilledPlugin
job.1.store.2.csv_time_scope=P1D
job.1.store.2.csv_time_slice=PT15M
job.1.store.2.csv_number_locale=de_AT
job.1.store.2.csv_file_path=exports/
job.1.store.2.csv_file_template=${job.start:YYYY-mm-dd}.csv
job.1.store.2.csv_header_template=Das ist der Header meiner CSV
job.1.store.2.csv_empty_line_template=${item.start:YYYY-mm-dd
HH:mm:ss};${item.end:YYYY-mm-dd HH:mm:ss};
job.1.store.2.csv_line_template=${item.start:YYYY-mm-dd
HH:mm:ss};${item.end:YYYY-mm-dd HH:mm:ss};${item.value:0.00}

activated_jobs=1

meta.job.1.data_type=Energy Consumption
meta.datastream.1.customer_id=1234
meta.datastream.2.customer_id=5678

```

Starten / Ausführen des Programms

Nach erfolgreicher Konfiguration kann das Programm über die `start.cmd` Datei gestartet werden. Sollte das Programm nicht ohnehin über das Terminal gestartet werden, öffnet sich beim Aufruf der Datei ein Terminal Fenster. Solange das Fenster **nicht geschlossen** wird, wird das Programm ausgeführt. Das Programm beinhaltet einen Scheduler, der je nach Konfiguration die Sync Jobs automatisch und periodisch startet. Dieser Scheduler verhindert es, dass das Programm terminiert.

Wird das Programm trotzdem unerwartet beendet, ist das ein Hinweis auf einen Fehler. Bitte senden Sie in diesem Fall den Inhalt der Datei `service.log` an ticket@linemetrics.com.