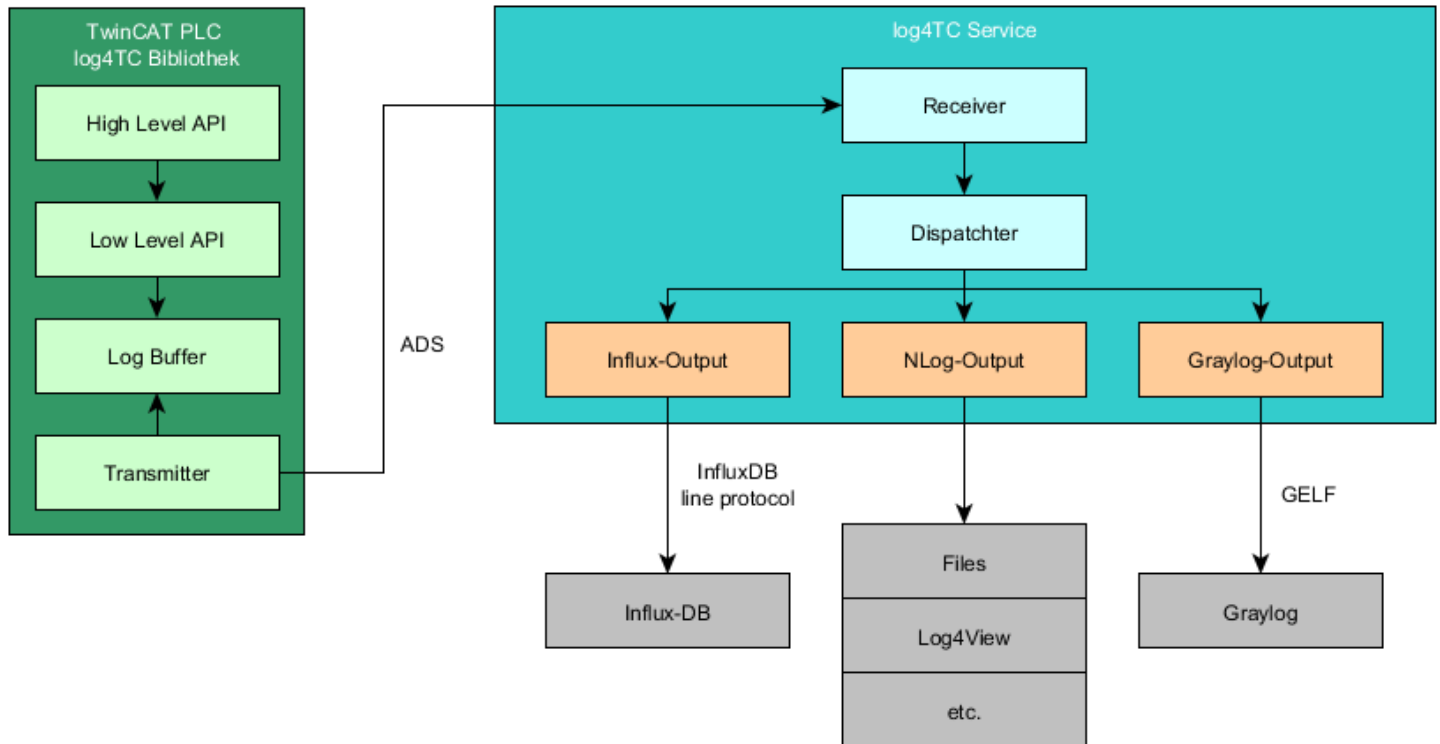


# mbc log4TC

Log4TC ist eine Erweiterung für TwinCAT3 von Beckhoff, um direkt aus der SPS Logmeldungen erzeugen zu können. Die Meldungen können transferiert, gefiltert, ausgewertet und an verschiedene Ausgaben weitergeleitet werden.

Log4TC besteht aus zwei Teilen, einer SPS-Bibliothek und einen Windows-Service.

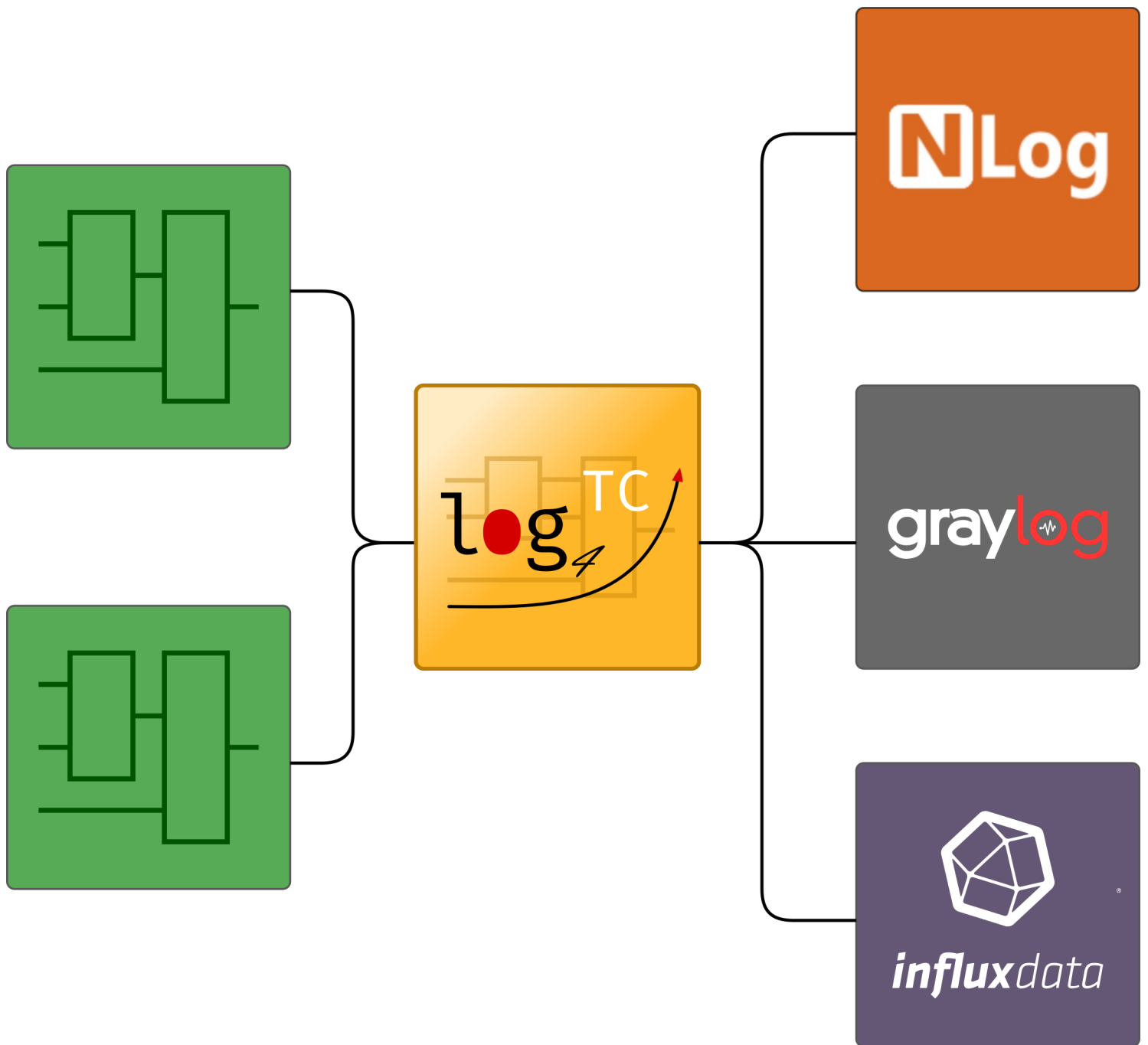


Der log4TC-Service wird normalerweise auf dem Rechner installiert, auf dem auch die SPS läuft, kann aber für bestimmte Einsatzzwecke auch auf einem anderen Rechner für mehrere Steuerung installiert werden.

## Features

- Einfache API für die integration in die SPS
- Strukturiertes Logging (<https://message templates.org/>)
- Unterstützung von Context-Eigenschaften auf verschiedenen Ebenen
- Performant und Modular
- Kostenlose Testversion verfügbar
- Lizenzierung über Beckhoff-Mechanismus in Dongle, Klemme oder PC
- Unbegrenzte Ausgabemöglichkeiten (Textdatei, Datenbank, Cloud, usw.)

## Ausgaben




Log4TC implementiert Ausgaben über ein Plugin-System. Standardmässig bei der Auslieferung ist die NLog-Ausgabe aktiv. Die Ausgabeplugins werden laufend erweitert, die nächsten Plugins sind Ausgaben für Graylog und InfluxDB.

Bei Bedarf können wir auch kundenspezifische Ausgaben erstellen.

## Typische Anwendungsfälle für log4TC

- Fehlertracking und Alarmierung
- Debugging von sporadischen Fehlern ohne Breakpoints
- Ablaufanalysen bei Problemen auch in Nachhinein
- Statistische Auswertungen, z.B. KPI

# Nächste Schritte

- [Download](#) 
- [Erste Schritte](#)
- [Referenz](#)

# Table of Contents

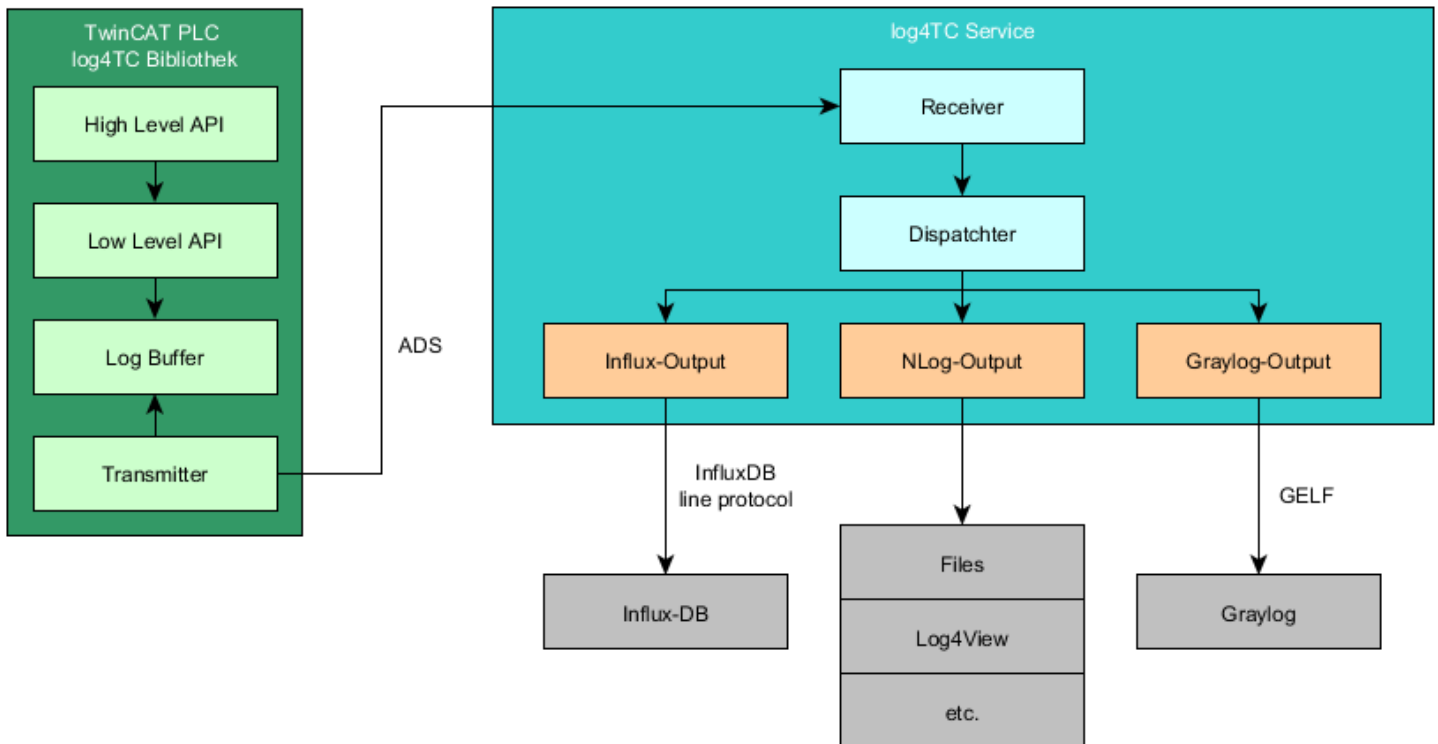
Erste Schritte .....	5
Referenzen .....	7
Changelog .....	9

# Erste Schritte

Diese geführte Tour stellt die grundlegenden Konzepte und Features von log4TC in einem kleinen zusammenhängenden Projekt vor.

## Vorraussetzungen

Damit log4Tc richtig benutzt werden kann, sollte der Aufbau bekannt sein. Es wird dabei unterschieden zwischen den Komponenten **log4TC TwinCat 3 Bibliothek** und dem **log4TC Service**.



Folgende Voraussetzungen haben die beiden Komponenten:

### log4TC TwinCat 3 Bibliothek

- [TwinCat 3 \(min. 4022.00\)](#)

### log4TC Service

- min Windows 7 SP1 / Windows Embedded Standard 2009
- [Microsoft .NET Framework \(Mindestends 4.6.1, empfohlen 4.8\)](#)
- [ADS Router - TC1000 | TC3 ADS](#)

Beim [Installieren](#) von log4TC wird eine Default-Konfigurationsdatei mit installiert. Für die nachfolgenden Beispiele wird davon ausgegangen, dass diese Konfiguration aktiv ist.

Für diese Einführung wird ausserdem vorausgesetzt, dass die SPS und der log4TC auf dem *gleichen* Rechner laufen (Testlizenz ist ausreichend). Dies ist keine Einschränkung von log4TC sondern eine Vereinfachung.

## Übersicht

Die Einführung geht schrittweise vor. Es wird empfohlen beim ersten Kontakt mit log4TC alle Schritte nacheinander selbst auszuprobieren.

1. [TwinCAT Projekt anlegen](#)
  2. [log4TC-Library hinzufügen](#)
  3. [Ausgabe einer einfachen Log-Meldung](#)
  4. [Ausgabe von Log-Meldungen mit Argumenten](#)
  5. [Benutzung von Loggern](#)
  6. [Integration von Context-Eigenschaften](#)
  7. [Log-Meldungen mit Log4View beobachten](#)
  8. [Protokollierung von strukturierten Werten](#)
- 

## Nächster Schritt

[TwinCAT Projekt anlegen](#)

# Werkzeuge rund um log4TC

## Anzeigen von Log-Meldungen

### Log4View

Die Anwendung [Log4View](#) von Prosa ist eine unserer Empfehlungen für die Anzeige von mittleren oder grossen Anzahl an Log-Meldungen. Selbst die kostenlose Variante ist für viele Zwecke ausreichend.

Wichtig bei der Benutzung von Log4View mit log4TC sind folgende Punkte:

### NLog Konfiguration

NLog muss für die log4j-XML Ausgabe konfiguriert werden. NLog kommt mit einem Standard-Layout `log4jxmlevent`, das prinzipiell diese Anforderung erfüllt. log4TC stellt aber eine erweiterte Variante `mbclog4jxmlevent` mit folgenden Verbesserungen zur Verfügung:

- Thread-Id enthält die SPS Task-ID
- Zusätzliches Properties um die Message zu formatieren

Eine Beispiel-Konfiguration könnte so aussehen:

```
<extensions>
  <add assembly="Mbc.Log4Tc.Output.NLog"/>
</extensions>

<targets>
  <target name="xmlLogFile"
    xsi:type="File"
    encoding="utf-8"
    fileName="${logdir}/log4tc.xml"
    <!-- evtl. weiter File Optionen -->
```

```
layout="${mbclog4jxmlevent:includeAllProperties=true:message=${message} [{mbc-all-event-
properties}]]}">
  </target>
```

### Logging

[Log4View](#) bietet sehr gute Selektionsmöglichkeiten auf dem Logger und dem Level. Es lohnt sicher daher diese beiden Konsequenz im Code einzusetzen.

Leider bietet [Log4View](#) bis jetzt noch keine Unterstützung von Context-Properties und/oder structured Logging.

# Notepad++

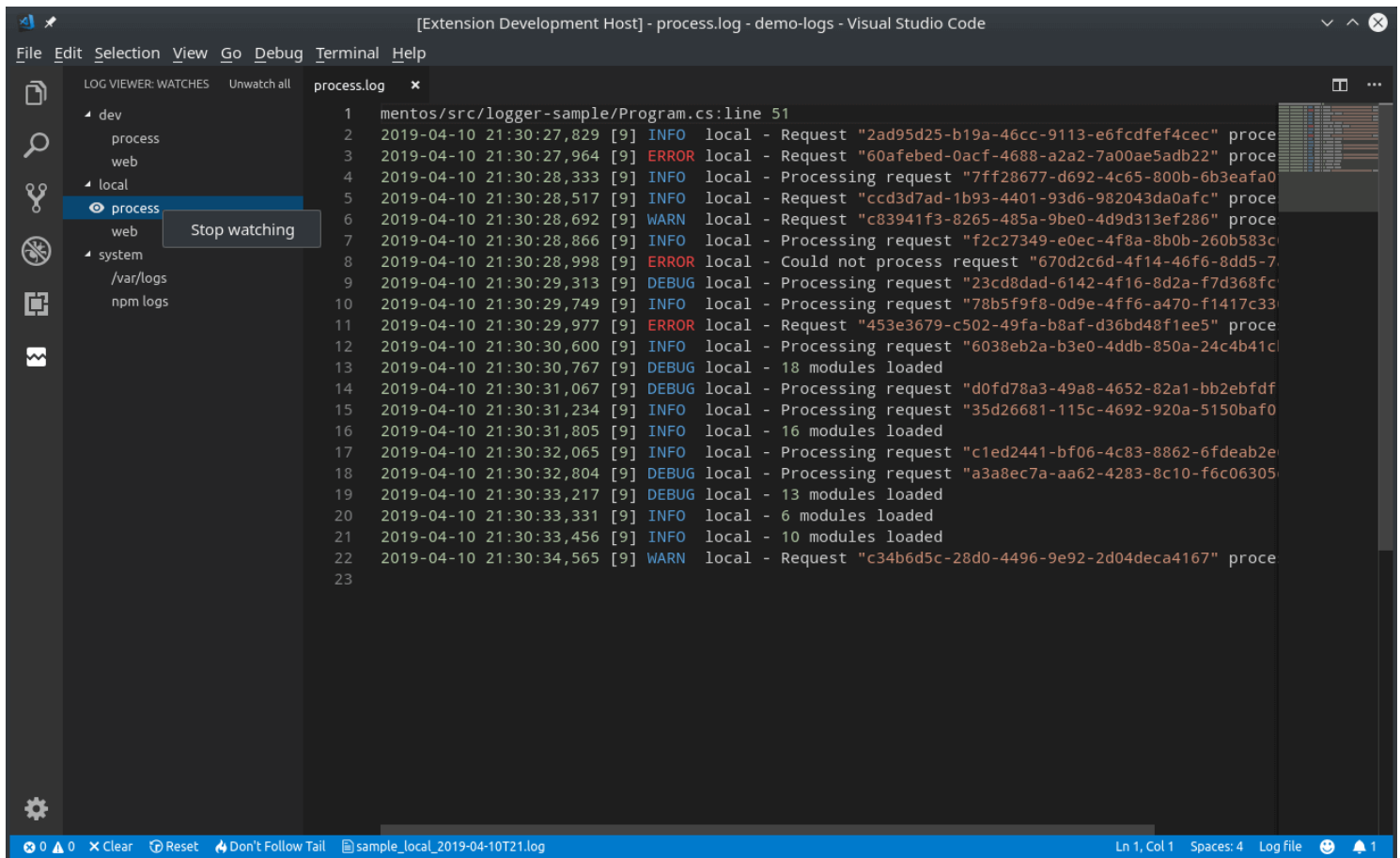
[Notepad++](#) bietet zwar keinen speziellen Modus für log4TC besitzt aber zwei Eigenschaften, die es interessant machen für einfache Logging Aufgaben:

- Im Menü "Ansicht" -> "Überwachen (tail -f)" kann Notepad++ angewiesen werden die Log-Datei laufend zu überwachen und bei Änderung automatisch einzulesen.
- Im Menü "Sprachen" kann nach eigenen Anforderungen ein Stil definiert werden, wie die einzelnen Blöcke in einem Log-File formatiert werden sollen

## Visual Studio Code

[Visual Studio Code](#) ist ähnlich wie Notepad++ eine universelle Plattform für Textverarbeitung. VS-Code bietet eine grosse Anzahl an Erweiterungen so z.B. auch für Log-File.

Eines davon *Log Viewer* bietet einfache Möglichkeiten für Log-Files:





# Changelog

## [VNext]

### Added

- Dokumentation steht nun auch als PDF zur Verfügung

### Fixed

- #15 - Links in Dokumentation korrigiert

## [24.01.18]

### Added

- NLog unterstützt nun auch die Ausgabe für Azure ApplicationInsight über das neue nlog Target `ApplicationInsightsTargetLog4Tc`.

### Security

- Update SQLClient

## [21.04.17]

### Added

- Neue ANY-Datentypen fürs Logging: TIME, LTIME, DATE, DATE\_AND\_TIME, TIME\_OF\_DAY, ENUM (numerisch), WSTRING

### Fixed

- Einige Startup Probleme behoben
- Setup of the library works now with TwinCat 4024 shell

### Changed

- Einige Logeinträge

## SPS-Library 0.2.1

- Lizenz Prüfung entfernt
- Library ist nun mbc\_Log4TC.library anstelle von mbc\_Log4TC.compiled-library
- E\_Scope nun public verwendbar

## SPS-Library 0.1.0

- Neue Funktion zum einfachen loggen mit ContextBuilder jedoch ohne Angabe des Loggers: `F_LogC`
- Erlauben `E_LogLevel` Wert als String zu konvertieren mit `{attribute 'to_string'}`
- 0-Copy Logging: Log-Meldungen werden direkt in den Sendebuffer geschrieben
- Neue ANY-Datentypen fürs Logging: TIME, LTIME, DATE, DATE\_AND\_TIME, TIME\_OF\_DAY, ENUM (numerisch), WSTRING

## [20.10.21]

### SPS-Library 0.0.6

- Lizenzierung für Windows-CE Geräte (keine OEM möglich)

### SPS-Library 0.0.8

- Lizenzierung Bugfix für Lizenzierungsklemme

## [20.10.15]

### Added

- Neuer Output für SQL-Datenbanken

### Changed

- `LogEntry` können jetzt als Collection im Output statt einzeln verarbeitet werden
- Influx-DB Output wurde angepasst, dass mehr als eine `LogEntry` gesendet wird

### Fixed

- Fehler im async-Handling beim Verarbeiten der Meldungen behoben; der Bug führte dazu, dass im Fehlerfall Exceptions nicht geloggt wurden
- NLog wurde initialisiert, auch wenn das Output-Plugin nicht konfiguriert wurde

## [20.07.28]

### Added

- Logging für Service aktiviert (`%ProgramData%/log4Tc/internal/service-*.log`)
- Konfiguration des log4TC Dispatcher (Ausgabe-Plugins)
- Neue Ausgabe für Influx-DB ( $\geq 1.8$ )
- Neue Ausgabe für Graylog

### Changed

- Setup angepasst, es ist nun eine Auswahl von Features möglich für die Szenarien PLC, PLC+Dev, Host

- Neue PLC-Library mit der Version 0.0.5

## Fixed

- Bugfix PLC: Context.AddString verwendet jetzt die korrekte Stringlänge
- Bugfix PLC: Log-Argumente wurden nicht korrekt übergeben (ab Argumente 3)
- Im Context überschreiben jetzt gleiche Namen den vorherigen Wert

## [20.05.06]

## Added

- Log4Tc Initial Version

### NOTE

All wichtigen Änderungen zu diesem Projekt werden in dieser Datei dokumentiert.

Das Release Versionierungsformat folgt dem Schema **Jahr.Monat.Tag.Patch**. Zum Beispiel: 20.05.06, 19.07.30.0, 19.07.30.1.

Es gibt folgende Änderungstypen:

- **Added** for new features.
- **Changed** for changes in existing functionality.
- **Deprecated** for soon-to-be removed features.
- **Removed** for now removed features.
- **Fixed** for any bug fixes.
- **Security** in case of vulnerabilities.