ToddySoft Connect OPC UA - Apache StreamPipes

Ein **Adapter** für Apache StreamPipes für OPC-UA-Server, auf Basis des ToddySoft-Connect-Treibers OPC UA. Er liest Variablen aus beliebigen OPC-UA-Servern, ausgewählt aus dem durchsuchten Adressraum des Servers. Jedes Abfrageintervall erzeugt in StreamPipes ein Ereignis mit den Werten der gewählten Variablen.

Der Adapter kommt als eigener **Erweiterungsdienst** (*extension service*): ein Container, der neben Ihrer StreamPipes-Installation läuft, sich dort anmeldet und in StreamPipes als Adapter *ToddySoft Connect OPC UA* erscheint.

> Die Bildschirmfotos zeigen die englische Oberfläche von StreamPipes, und die Bezeichnungen der Felder und Menüs werden hier englisch genannt. Bei deutscher Oberfläche lauten einige Menüpunkte anders, zum Beispiel *Service-Konfigurationen* für *Service configurations*, *Aktualisieren* für *Update* und *Neu laden* für *Reload*.

Voraussetzungen

- **Apache StreamPipes 0.99**, mit Docker installiert (der Standard-Installer). Der Dienst tritt dem Docker-Netz der Installation bei.
- **Linux auf dem Rechner mit StreamPipes**, amd64 oder arm64, mit einer Route zu den Geräten. Docker Desktop (macOS, Windows) betreibt Container in einer virtuellen Maschine: Adapter funktionieren mit *Enter address*, aber keine Suche erreicht das Anlagennetz.
- Eine **ToddySoft-Lizenz für OPC UA** für den produktiven Einsatz. Ohne sie läuft der Dienst mit seiner mitgelieferten Testlizenz, bis diese abläuft.

Installation

Das Image laden

Der Dienst wird als Image-Archiv geliefert, eines je Architektur, in der Freigabe, über die Sie Ihre Lizenz und Downloads erhalten: `streampipes-opcua-<Version>-<Architektur>.tar.gz`, für `amd64` oder `arm64`. Verwenden Sie das Archiv der Ihnen gelieferten Version und der Architektur des Rechners (`uname -m`: `x86_64` ist amd64, `aarch64` ist arm64).

```
docker load < streampipes-opcua-1.3.0-amd64.tar.gz
```

> **Die Images liegen weder auf Docker Hub noch in einer anderen öffentlichen Registry**, und dasselbe gilt für den Discovery-Agenten: Jedes Image erreicht Sie über diesen einen Weg. Wir planen den Umstieg auf eine private Registry mit einem Zugang je Kunde, sodass `docker pull` und Updates ohne Archive funktionieren; bis dahin laden Sie jede neue Version auf dieselbe Weise.

Den Dienst starten

Starten Sie den Dienst im Docker-Netz Ihrer StreamPipes-Installation (`spnet` beim Standard-Installer; `docker network ls` zeigt es):

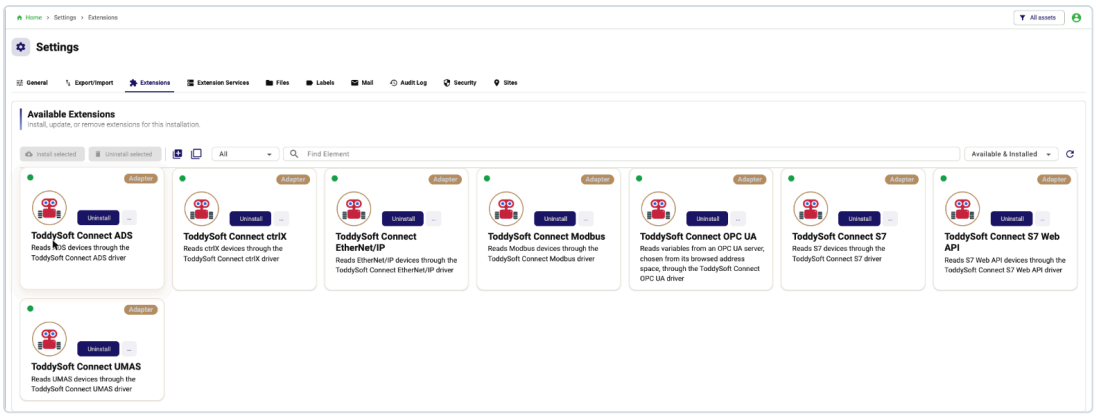
```
docker run -d --name toddysoft-opcua --restart unless-stopped \
  --network spnet \
  -e SP_HOST=toddysoft-opcua -e SP_PORT=8005 \
  -e SP_CLIENT_SECRET=<das SP_CLIENT_SECRET Ihrer Installation> \
  -e SP_PRIORITIZED_PROTOCOL=kafka \
  toddysoft/streampipes-opcua:1.3.0
```

- **`SP_HOST`** muss der eigene Name des Containers sein: Unter diesem Namen ruft StreamPipes den Dienst zurück.
- **`SP_CLIENT_SECRET`** muss der Wert sein, mit dem der StreamPipes-Kern läuft (die `.env` des Installers). Mit einem anderen Wert kann sich der Dienst nicht anmelden.
- **`SP_PRIORITIZED_PROTOCOL`** ist der Broker Ihrer Installation: `kafka` beim Standard-Installer.
- Lassen Sie den Dienst nur im Netz der Installation. Dort erreicht er Kern und Broker über ihre Namen; mit Host-Netzwerk gingen sie verloren. Die Geräte erreicht er über das Routing des Rechners, die Suche über den Discovery-Agenten (siehe Die Geräte erreichen).
- Dateien, die der Adapter braucht - ein Keystore, ein Trust Store -, binden Sie mit `-v` ein, zum Beispiel `-v /opt/toddysoft/certs:/certs:ro`, und geben sie im Formular als `/certs/<Datei>` an.

`docker logs toddysoft-opcua` sollte innerhalb einer Minute zeigen, dass sich der Dienst bei StreamPipes angemeldet hat.

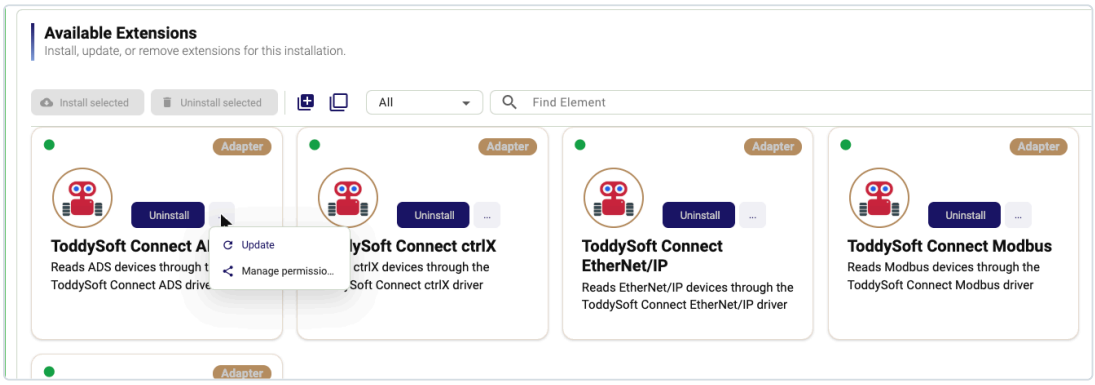
Den Adapter in StreamPipes installieren

Sobald sich der Dienst angemeldet hat, installieren Sie seinen Adapter einmalig: **Settings** → **Extensions**, *ToddySoft Connect OPC UA* suchen und **Install** drücken. Danach erscheint er unter **Connect** → **New Adapter**.



Aktualisieren

1. Laden Sie das neue Archiv und starten Sie die neue Version anstelle des alten Containers (`docker rm -f toddysoft-opcua`, dann das `docker run` oben mit der neuen Version).
2. **Aktualisieren Sie den installierten Adapter**: Settings → Extensions → **"..."** des Adapters → **Update**. StreamPipes behält die bei der Installation gespeicherte Beschreibung des Adapters; ohne diesen Schritt erscheint das Formular der vorigen Version, und neue Einstellungen fehlen.

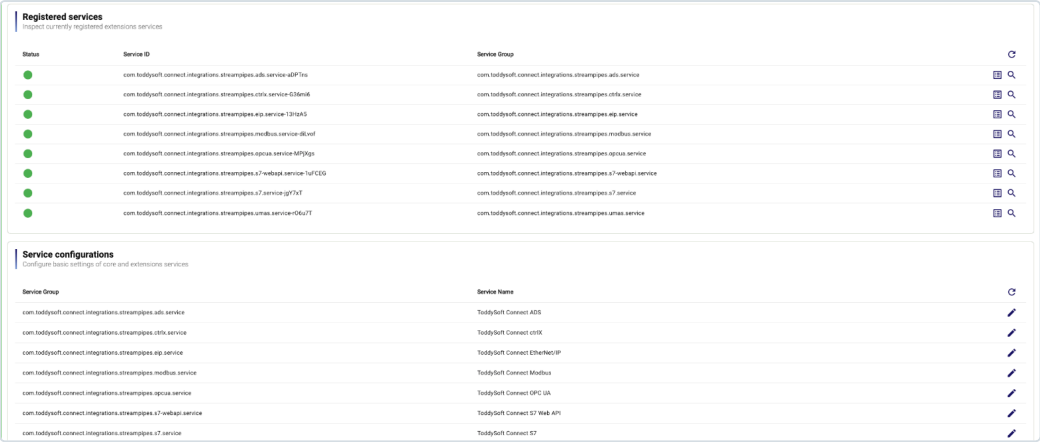


Mit der alten Version gespeicherte Adapter laufen nach einem Update weiter.

Lizenz

Der Treiber liest seine Lizenz aus der Konfiguration des Dienstes in StreamPipes:

1. Öffnen Sie **Settings** → **Extension Services**. *Registered services* listet die laufenden ToddySoft-Dienste; **Service configurations** darunter enthält einen Eintrag je Dienst.



2. Drücken Sie den Stift neben *ToddySoft Connect OPC UA*. Der Eintrag hat ein Feld, `licence-opcua`.



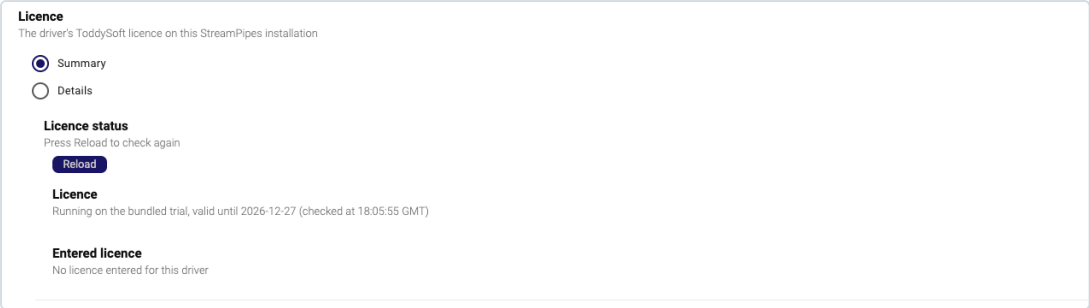
3. Fügen Sie die Lizenzdatei **Base64-kodiert** ein: `base64 -i opcua.lic` (macOS) oder `base64 -w0 opcua.lic` (Linux) gibt sie aus.

4. **Klicken Sie nach dem Einfügen noch einmal in das Feld** und drücken Sie dann **Update**. StreamPipes übernimmt den Wert des Feldes erst bei diesem Klick; wer direkt nach dem Einfügen speichert, speichert den vorigen Wert, meist einen leeren.

Die Lizenz wirkt ab dem nächsten Durchsuchen oder Adapterstart; ein Neustart ist nicht nötig. Ohne Lizenz, oder mit einer, die der Treiber ablehnt, gilt die mitgelieferte Testlizenz bis zu ihrem Ablauf. Der Wert wird im Formular verdeckt angezeigt, in der Datenbank von StreamPipes aber wie eingegeben gespeichert; beschränken Sie daher, wer StreamPipes verwalten darf.

Die Lizenz prüfen

Jedes Adapterformular beginnt mit einem Block **Licence**, zusammengeklappt auf die **Summary**: ob der Treiber lizenziert ist und durch welche Lizenz, bis wann, und was aus der eingegebenen Lizenz geworden ist. **Details** zeigt alles, was der Treiber über die verwendete Lizenz meldet: ihre Prüfsumme (die sie kennzeichnet), auf wen sie lautet, wie lange sie gilt, bis wann Support besteht, und ihre Stufe (Tier). Drücken Sie nach einer Änderung der Lizenz **Reload** im Block.



der Block zeigt	Bedeutung	was zu tun ist
Licensed by the licence entered in the service configuration · Entered licence: In use	Ihre Lizenz gilt	nichts
Running on the bundled trial · No licence entered for this driver	keine Lizenz eingegeben; die Testlizenz gilt bis zum angezeigten Datum	Lizenz eingeben (siehe oben)
Entered licence: Not in use: the entered value cannot be read as a licence (position N)	das Feld enthält keine Base64-kodierte Lizenzdatei, meist mitkopierte Zeichen	die kodierte Datei erneut einfügen, ohne etwas davor oder dahinter
Entered licence: Not in use: refused by the driver	die Lizenz ist echt, deckt dies aber nicht ab: anderer Treiber, andere Plattform, abgelaufen	Lizenz prüfen; der Grund steht im Log des Dienstes (`blob rejected for driver`)
Licensed by a licence file	eine Lizenzdatei im Lizenzverzeichnis des Dienstes gilt	nichts
Attention: Expires in N days / Support ends in N days	die Lizenz oder die von ihr abgedeckten Treiberversionen enden bald	rechtzeitig verlängern
Attention: Expired on and running in its grace period until	die Lizenz ist abgelaufen; ToddySoft Connect akzeptiert sie noch 30 Tage (Nachfrist)	jetzt verlängern: nach diesem Datum wird die Lizenz abgelehnt

Felder, die diese Treiberversion noch nicht meldet (was die Lizenz abdeckt, wann sie ausgestellt wurde, ihre Beschreibung), erscheinen als *not reported by this driver version*. Die Lizenz selbst zeigt der Block nie.

Die Geräte erreichen

Ein Dienst im Container sieht nur seine Container-Netze. Verbinden, Durchsuchen und Lesen laufen über das Routing des Rechners und funktionieren, solange der Rechner eine Route zum Gerät hat. **Die Suche** nicht: Broadcasts, Multicast und PROFINET-DCP verlassen das Netz des Containers nie, allein findet die Suche im Anlagennetz also nichts. Es gibt zwei Wege, ihr das Anlagennetz zu geben.

Der ToddySoft-Discovery-Agent (empfohlen)

Der Agent ist ein kleiner Container, der einmal je Rechner mit dem Netzwerk des Hosts läuft und die Anlagennetze für jeden ToddySoft-Dienst durchsucht, der danach fragt. Er sucht nur: Der Adapter verbindet sich weiterhin aus seinem eigenen Container. Er kommt als Archiv neben dem dieses Dienstes, `discovery-agent-<Version>-<Architektur>.tar.gz`; verwenden Sie den Agenten derselben Version.

```
docker load < discovery-agent-1.3.0-amd64.tar.gz
docker run -d --name toddysoft-discovery-agent --restart unless-
        stopped \
  --network host --cap-add NET_RAW \
  toddysoft/discovery-agent:1.3.0
```

- **Am Dienst ist nichts einzustellen.** Bei jeder Suche sucht er den Agenten am Gateway seines Container-Netzes, unter Docker also am Rechner selbst. Ein später gestarteter Agent wird beim nächsten Reload gefunden. Die Statuszeile der Suche lautet *via discovery agent*, wenn der Agent geantwortet hat, *no discovery agent found · container networks only*, wenn keiner läuft, oder nennt den Grund, aus dem der Agent abgelehnt hat.
- **Lizenz.** Der Dienst fragt nur nach OPC UA-Geräten und sendet die Lizenz, mit der sein Treiber läuft, über eine verschlüsselte Verbindung; der Agent prüft sie, verwendet sie für diese eine Suche und behält nichts.
- **Sicherheit.** Der Agent lauscht nur auf den Docker-Netz-Adressen des Rechners (TCP 47900) und nimmt Verbindungen nur aus den Container-Netzen an; geben Sie diesen Port für sie frei, wenn der Rechner eine Firewall hat. Für mehr Schutz setzen Sie ein gemeinsames Geheimnis: `-e TODDYSOFT_AGENT_SECRET=<Geheimnis>` am Agenten und `-e TODDYSOFT_DISCOVERY_AGENT_SECRET=<Geheimnis>` am Dienst.
- **Finden ist nicht Erreichen.** Ein Gerät, das der Agent in einem Netz findet, in das der Rechner nicht routen kann, wird angezeigt und lässt sich dann nicht verbinden.

Ein zweites Netz an der Anlagenschnittstelle

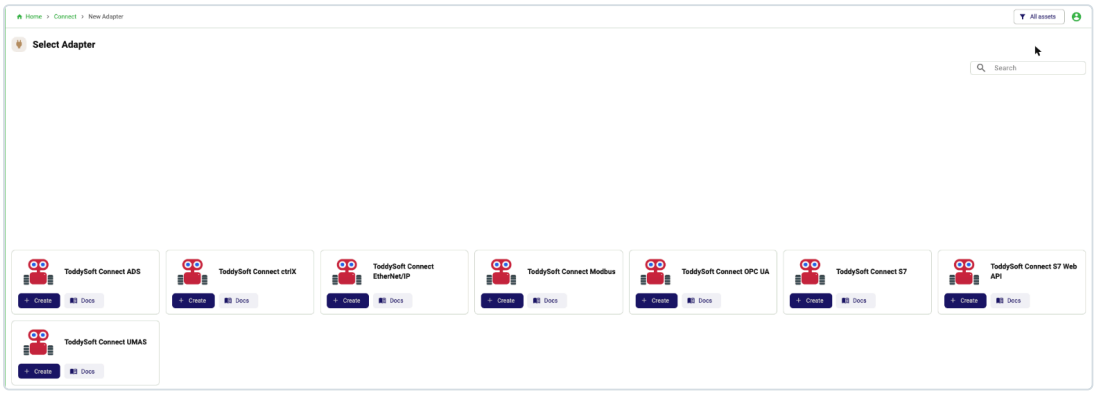
Alternativ verbinden Sie den Dienst mit einem macvlan-Netz an der Anlagenschnittstelle, im Subnetz der Geräte, als **zweites** Netz (`spnet` bleibt):

```
docker network create -d macvlan --subnet 192.168.23.0/24 --gateway
        192.168.23.1 -o parent=eth1 plant
docker network connect --ip 192.168.23.250 plant toddysoft-opcua
```

Das braucht jeder Dienst, einmal je Anlagensubnetz. Setzen Sie `-e TODDYSOFT_DISCOVERY_AGENT=off` an einem Dienst, dessen einziges Netz ein macvlan ist, damit er den Agenten nicht am Router der Anlage sucht.

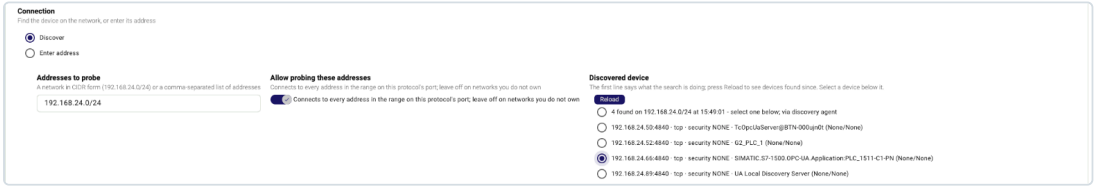
Einen Adapter anlegen

Öffnen Sie **Connect** → **New Adapter** und drücken Sie **Create** bei *ToddySoft Connect OPC UA*.



Das Gerät finden

OPC UA-Geräte melden sich nicht selbst; sie werden durch **Ausprobieren von Adressen** gefunden: Der Dienst verbindet sich mit jeder Adresse eines Bereichs auf dem Port des Protokolls. Das geschieht nur mit Ihrer Zustimmung.



1. Tragen Sie unter **Connection** → **Discover** die **Addresses to probe** ein: ein Netz wie `192.168.24.0/24` oder eine durch Kommas getrennte Liste von Adressen.
2. Schalten Sie **Allow probing these addresses** ein. Lassen Sie es in Netzen aus, die nicht Ihnen gehören.
3. Drücken Sie **Reload** neben **Discovered device**.

Bevor beides gesetzt ist, wird nichts abgefragt. Mit dem Discovery-Agenten gehen Bereich und Zustimmung unverändert an den Agenten; eine Zustimmung mit leerem Bereich fragt die Anlagennetze des Agenten ab.

Die erste Zeile von **Discovered device** sagt, was die Suche gerade tut: Sie läuft bis zu 25 Sekunden im Hintergrund, und **Reload** zeigt die seither gefundenen Geräte. Wählen Sie ein Gerät unterhalb dieser Zeile. Wird nichts gefunden, nennt die Zeile die durchsuchten Netze und jeden abgelehnten Bereich. Eine beendete Suche wird eine Minute lang wiederverwendet.

Sicherheit

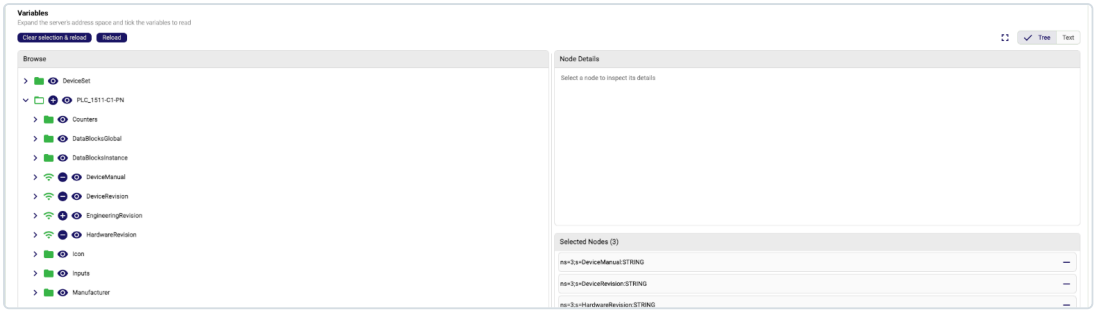
Die Verbindung ist derzeit **ungesichert und anonym**: Nur Endpunkte, die dieser Adapter ohne Sicherheit öffnen kann, werden angezeigt und verwendet. Gesicherte Kanäle und Benutzer-Zugangsdaten sind geplant. Verwenden Sie ihn nur in einem vertrauenswürdigen Netz oder mit einem Server, der zum Lesen einen ungesicherten Endpunkt anbietet.

Einstellungen

Einstellung	Bedeutung
Connection	*Discover* (Server suchen) oder *Enter address* (die Felder unten selbst eintragen)
Addresses to probe	Netz (`192.168.24.0/24`) oder durch Kommas getrennte Adressen; siehe oben
Allow probing these addresses	Zustimmung, jede Adresse des Bereichs anzusprechen
Host, Port, Endpoint path	Wo der Server ist: `opc.tcp://host:4840/pfad` ergibt Host, Port und Pfad
Polling interval (ms)	Wie oft die gewählten Variablen gelesen werden; jeder Lesevorgang ergibt ein Ereignis
Add timestamp	Fügt jedem Ereignis das Feld `timestamp` (Zeitpunkt des Lesens) hinzu; siehe unten
Tags	Was gelesen wird; siehe unten

Der Link **Documentation** oben im Formular öffnet eine Kurzfassung dieser Seite in StreamPipes.

Die Variablen wählen



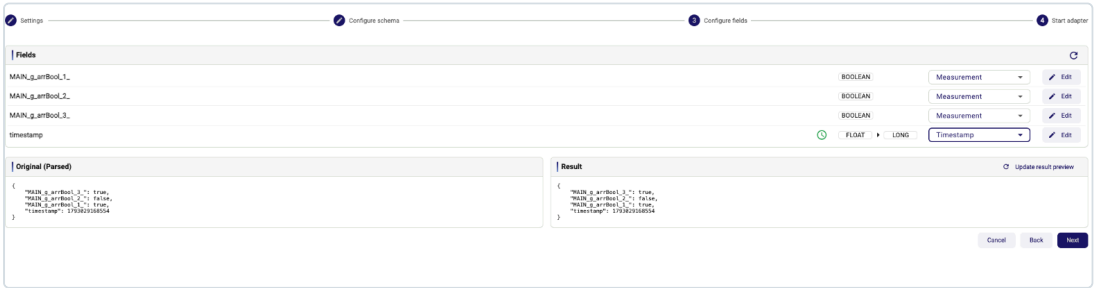
Der Adressraum des Servers wird **Ebene für Ebene** durchsucht, während Sie ihn aufklappen. Klappen Sie Ordner und Objekte auf, um Variablen zu finden; nur Variablen mit bekanntem Datentyp lassen sich auswählen. Jede wird zu einem Feld, benannt nach ihrer Node-ID (`ns=2;s=Demo.Dynamic.Double` wird zu `Demo_Dynamic_Double`).

Der **Text**-Modus des Baums bearbeitet die Auswahl direkt, eine Adresse je Zeile - praktisch, um eine Liste von Adressen einzufügen. StreamPipes schreibt eigene Kommentarzeilen in diesen Editor (`# Provide OPC UA Node IDs …`); Zeilen, die mit `#` beginnen, werden ignoriert.

Ereignisse und der Zeitstempel

Drücken Sie **Next**. StreamPipes liest die Variablen einmal und zeigt das Beispielereignis; danach erzeugt jede Abfrage ein solches Ereignis, mit einem Feld je lesbarer Variable und **`timestamp`**, dem Zeitpunkt des Lesens in Millisekunden seit der Epoche.

Der Adapter kann StreamPipes nicht mitteilen, welches Feld der Zeitstempel ist. Setzen Sie unter **Configure fields** die Rolle von `timestamp` auf **Timestamp** und den Typ auf **Long**:



- StreamPipes 0.99 typisiert im abgeleiteten Schema **jede Zahl als Gleitkommazahl**. Die Ereignisse selbst tragen die exakten Werte; ändern Sie den Typ eines Feldes hier, wo es darauf ankommt.
- Eine Variable, die selbst `timestamp` heißt, erscheint als `timestamp_tag`.
- Mit ausgeschaltetem **Add timestamp** entfällt das Feld, für eine Pipeline, die ihren eigenen setzt.

Benennen Sie den Adapter anschließend und starten Sie ihn unter **Start adapter**.

Fehlersuche

- **Das Formular sieht aus wie eine ältere Version, oder eine hier beschriebene Einstellung fehlt**: Aktualisieren Sie den installierten Adapter (Settings → Extensions → "…" → Update).
- **Der Variablenbaum zeigt einen Fehler**: Die Meldung nennt die Ursache - eine nicht erreichbare Adresse, eine abgelehnte Anmeldung, eine fehlende Lizenz. Lautet sie nur *An unknown error has occurred*, steht das Nähere im Protokoll des Dienstes: `docker logs toddysoft-opcua`.
- **Die Suche findet nichts**: Lesen Sie ihre Statuszeile. *Container networks only* heißt, dass weder der Discovery-Agent noch ein Anlagennetz vorhanden ist (siehe Die Geräte erreichen).
- **Die Lizenz wird nicht übernommen**: Klicken Sie nach dem Einfügen in das Feld, dann Update. Eine Lizenz, die der Treiber ablehnt, wird als `blob rejected for driver` protokolliert, und die Testlizenz gilt; der Block Licence im Adapterformular zeigt, welche Lizenz gilt und warum Ihre nicht (siehe Die Lizenz prüfen).
- **Der Dienst erscheint nicht unter Extensions**: Prüfen Sie `docker logs toddysoft-opcua`; meist stimmt `SP_CLIENT_SECRET` oder das Netz nicht.
- **Nach einer Änderung nur des Endpunktpfads zeigt der Baum noch den alten Server**: Drücken Sie **Reload** neben dem Variablenbaum; StreamPipes lädt ihn für dieses Feld nicht neu.

Versionshistorie

1.3.0 — 2026-11-13

Basiert auf ToddySoft Connect 1.3.0.

- **Neu:** die erste Version des OPC UA-Adapters für Apache StreamPipes 0.99 - Gerätesuche, Variablenbaum, Abfragen als StreamPipes-Ereignisse, die Lizenz in der Dienstkonfiguration samt ihrem Status in jedem Adapterformular und die Suche aus Containern über den ToddySoft-Discovery-Agenten.