Was ist PLCcom Opc Ua Sdk?

PLCcom Opc Ua Sdk stellt **.NET- und Java-Entwicklern** eine leistungsfähige Grundlage bereit, um OPC UA Client- und Server-Funktionalitäten schnell und kontrolliert in eigene Anwendungen zu integrieren.

Typische OPC UA Aufgaben wie Verbindungsaufbau, Lesen, Schreiben, Subscriptions, Events, Methoden oder die Bereitstellung eigener Server-Adressräume sind bereits als komfortable Funktionen gekapselt. Dadurch lassen sich auch anspruchsvolle OPC UA Lösungen **ohne tiefes OPC UA Spezialwissen** sauber und effizient umsetzen.
Das SDK unterstützt die aktuelle **OPC UA Spezifikation 1.05** und ist auf schnelle, robuste Integration ausgelegt.
Neben der klassischen NodeId-Adressierung unterstützt das SDK optional auch den Zugriff über **lesbare Browse-Pfade** wie **Objects.Plant.Machine1.Temperature**. Dadurch können Anwendungen verständlicher und wartbarer umgesetzt werden, ohne ausschließlich mit technischen NodeIds wie ns=2;i=1001 arbeiten zu müssen.
Für professionelle Server- und Client-Anwendungen stellt PLCcom Opc Ua Sdk flexible Schnittstellen bereit: eigene Zertifikatsvalidierung, eigene Benutzer- und Rechtekonzepte bis auf Node-Ebene sowie eigene Datenspeicher für Historical Data und Historical Events.
Damit ist das SDK eine flexible Grundlage für individuelle OPC UA Lösungen – von HMI-/SCADA bis hin zu Gateways, Diagnosewerkzeugen, Prüfständen und branchenspezifischer Automatisierungssoftware.
Die Komponente ist nach den Vorgaben der Cyberresilienz-Verordnung EU 2024/2847 (CRA) zertifiziert.

Technischer Kurzüberblick

Herausgeber	Indi.An GmbH (Copyright ©)
System	SDK/Library für OPC UA Client-, Server- und PubSub-Kommunikation
Aktuelle Version	10.x
Umgebung	.net und Java (je nach Auslieferungspaket)
OPC UA Spezifikation	ausgerichtet auf OPC UA Spezifikation 1.05
Betriebsarten	Client SDK, Server SDK, PubSub Add-on oder kombinierter Betrieb innerhalb derselben Host-Anwendung
Transport	OPC UA Binary über TCP (opc.tcp), OPC UA over https (opc.https / https) sowie OPC UA PubSub über UDP und MQTT. http wird technisch unterstützt, ist für produktive Umgebungen jedoch nicht empfohlen
Security	SecurityMode Sign / SignAndEncrypt, moderne RSA-Policies; .net zusätzlich ECC-Policies je nach Plattform/Build. PubSub unterstützt TLS-basierte MQTT-Verbindungen und PKI-basierte Zertifikatsprüfung.
CRA	Die Komponente ist gemäß **EU-Cyberresilienz-Verordnung 2024/2847 (CRA) zertifiziert**. Technische Dokumentation, SBOM und Vulnerability-Scan-Prozess sind Teil des Security-Konzepts.

Codebeispiele und Workshops

Die Beispiel-Repositories sind nicht nur kurze Code-Snippets, sondern als zusammenhängende Workshops aufgebaut. Sie führen schrittweise durch typische Client- und Server-Aufgaben und eignen sich dadurch besonders gut zur Einarbeitung neuer Entwickler.

Bereich	Inhalt
.net Repository	C#- und VB-Workshops für Client-, Server- und PubSub-Szenarien, inklusive Data Access, Alarme, Historical Access, PubSub über UDP/MQTT, NodeSet Import, Logging und Reverse Connect.
Java Repository	Java-Workshops für Client-, Server- und PubSub-Szenarien, inklusive Data Access, Alarms & Conditions, Historical Data/Events, PubSub über UDP/MQTT, komplexe Datentypen, NodeSet2 Import, Logging und Reverse Connect.
Onlinehilfe	API-Dokumentation, Klassenreferenz und technische Detailinformationen für die jeweilige .net- bzw. Java-Version.
Auslieferungspaket	Enthält zusätzliche Hilfen, Beispiele und Dokumentation passend zur jeweiligen Auslieferung.

https://github.com/Indi-An/PLCcom-OpcUaSdk-examples-dotnet

https://github.com/Indi-An/PLCcom-OpcUaSdk-examples-java

Lizenzierung per Lizenzschlüssel

Für den Betrieb ist ein Lizenzschlüssel erforderlich (User/Serial). Trial-Schlüssel sind verfügbar.

Hinweis: Einen 14-Tage trial-Key können Sie über unsere Website https://www.indi-an.com/de/plccom/opc-ua-sdk/opcua-download/ generieren.

Systemvoraussetzungen

PLCcom Opc Ua Sdk ist je nach Auslieferungspaket eine reine .net- oder Java-Library. Spezielle Hardwareanforderungen bestehen nicht; maßgeblich sind die Anforderungen der Host-Anwendung, der gewählten Runtime und der Zielumgebung.

.net Framework	.net Framework 4.7.2 und 4.8 oder neuer (.net-Framework-Pakete).
.net / .net Standard	.net 5, .net 6 und .net 7 sind über .net Standard 2.1 abgedeckt. Zusätzlich stehen direkte Pakete für .net 8, .net 9 und .net 10 zur Verfügung.
.net Einbindung	Einbindung als NuGet-Paket PLCcom.Opc.Ua.Sdk; das optionale PubSub Add-on wird versionsgleich bereitgestellt. Download: https://www.indi-an.com/de/plccom/opc-ua-sdk/opcua-download/
.net Entwicklung	Visual Studio 2022 oder neuer für Beispielprojekte empfohlen.
Java Runtime	Java JDK/JRE/OpenJDK 8 oder neuer; getestet bis Java 26.
Java Einbindung	Einbindung als Maven-Artefakt com.indi-an.plccom:plccom-opc-ua-sdk; das optionale PubSub Add-on com.indi-an.plccom:plccom-opc-ua-pubsub wird versionsgleich bereitgestellt. Download: https://www.indi-an.com/de/plccom/opc-ua-sdk/opcua-download/
Java Entwicklung	Maven und eine aktuelle Java-IDE wie IntelliJ IDEA, Eclipse, NetBeans oder Visual Studio Code mit Java-Erweiterungen empfohlen.

Betriebssysteme	.net-Framework-Varianten sind typischerweise Windows-orientiert. Moderne .net- und Java-Varianten können entsprechend der jeweiligen Runtime plattformübergreifend eingesetzt werden.

Bereitstellung / Integration

Direkte Einbindung	Das SDK wird direkt in die Host-Anwendung eingebunden. Die Anwendung steuert Lebenszyklus, Verbindungsaufbau, Security-Konfiguration und Datenmodell.
.net Lieferung	a. NuGet-Paket <PackageReference Include="PLCcom.Opc.Ua.Sdk" Version="10.*" /> b. optionales PubSub Add-on versionsgleich zum SDK c. optionales Auslieferungspaket: https://www.indi-an.com/de/plccom/opc-ua-sdk/opcua-download/
Java Lieferung	a. Maven-Artefakt com.indi-an.plccom:plccom-opc-ua-sdk b. optionales Maven-Artefakt com.indi-an.plccom:plccom-opc-ua-pubsub, versionsgleich zum SDK c. optionales Auslieferungspaket: https://www.indi-an.com/de/plccom/opc-ua-sdk/opcua-download/
Keine OPC Classic Altlasten	Keine COM-Registrierung, keine DCOM-Konfiguration und keine separate OPC-Classic-Infrastruktur erforderlich.
Client / Server / PubSub	Client-, Server- und PubSub-Funktionalität können separat oder gemeinsam innerhalb einer Anwendung verwendet werden.
Nicht standalone	PLCcom Opc Ua Sdk ist eine Library. Es ist kein eigenständiger OPC UA Server/Client, sondern ein Integrationsbaustein für eigene Software.

Architektur und Betriebsarten

Client Mode	Host-Anwendung -> PLCcom Opc Ua Sdk -> OPC UA Server. Typisch für HMI/SCADA, Prüfstände, Gateways, Diagnose- und Auswertesoftware.
Server Mode	OPC UA Client -> PLCcom Opc Ua Sdk -> Host-Anwendung / Datenmodell. Typisch für eigene Maschinen-, Linien-, Gateway- oder Simulationsserver.
Combined Mode	Client- und Server-Funktionalität parallel in derselben Anwendung, z. B. als Gateway zwischen vorhandenen OPC UA Servern und einem eigenen, vereinheitlichten Adressraum.
Adressraum	Serverseitig können Objekte, Variablen, Arrays, Properties, Methoden, Alarme, Ereignisse, mehrere Namespaces und importierte NodeSet2-Modelle bereitgestellt werden.

Funktionen OPC UA Client SDK

Das Client SDK stellt komfortable High-Level-Funktionen bereit, damit typische OPC UA Aufgaben schnell umgesetzt werden können, ohne die vollständige Komplexität des OPC UA Stacks manuell abzubilden.

Thema	Technischer Umfang
Discovery / Verbindung	Server und Endpoints ermitteln, Endpoint auswählen, Session aufbauen, Benutzer- oder Zertifikatsauthentifizierung verwenden.
Data Access	Lesen und Schreiben von Werten per NodeId oder Browse-Pfad, Batch-Operationen, Read/Write von Attributen.
Subscriptions	Monitoring von Variablen über Subscriptions und Monitored Items, inklusive DataChange- und Event-Szenarien.
Methoden	OPC UA Methoden aufrufen, auch mit strukturierten Eingabe- und Ausgabeparametern.
Alarme / Events	Alarme und Ereignisse abonnieren, Alarm-Liste aufbauen, Acknowledge, Confirm und Kommentare verarbeiten.
Historical Access	Historical Data lesen, Werte zu Zeitpunkten auslesen, Rohwerte und verarbeitete Werte verwenden, Historical Events lesen/überwachen.
Performance	Batch-Zugriffe, Registered Read/Write für häufig genutzte Knoten, Keep-Alive-Überwachung und klare Connection-Events.
Reverse Connect	Verbindungen in eingeschränkten Netzwerkarchitekturen, z. B. wenn klassische eingehende Verbindungen durch Firewalls erschwert sind.

Funktionen OPC UA Server SDK

Mit dem Server SDK kann eine eigene .net- oder Java-Anwendung einen OPC UA Server bereitstellen. Die Host-Anwendung liefert das Datenmodell, die Werte, die Security-Entscheidungen und bei Bedarf die historische Persistenz.

Thema	Technischer Umfang
Adressraum	Ordner, Objekte, Variablen, Arrays, Properties, Methoden, mehrere Namespaces und dynamische Knoten zur Laufzeit.
Data Access Server	Eigene Variablen und Werte bereitstellen, aktualisieren und über OPC UA Clients lesbar/schreibbar machen.
Methoden	Eigene Servermethoden mit Callback-Logik veröffentlichen und aus OPC UA Clients heraus aufrufen lassen.
Typen / Modelle	Custom Structured Data Types, komplexe Datentypen, Custom ObjectTypes/VariableTypes sowie NodeSet2 XML Import.
Alarms & Conditions	Alarmmodelle mit Zuständen, Limit-/Diskret-/Dialog-Alarmen, Acknowledge, Confirm, Kommentaren und Event-Ausgabe.
Historical Server	Historische Werte und historische Ereignisse bereitstellen; Persistenz über austauschbare Store-Interfaces anbinden.
Security / Benutzer	UserTokenPolicies, Benutzer-/Passwortprüfung, Zertifikatsauthentifizierung, Rollen und eigene Credential-/Permission-Validatoren.
Diagnose / Betrieb	Logging, Connection-Events, Operation Limits, Session-Limits und konfigurierbare Security Policies.

OPC UA PubSub Add-on

Das optionale PubSub Add-on erweitert das SDK um OPC UA PubSub Publisher- und Subscriber-Funktionalität. Es baut auf dem vorhandenen OPC UA Stack auf und stellt High-Level-Konfigurationen für typische PubSub-Szenarien bereit.

Thema	Technischer Umfang
Publisher	OPC UA PubSub DataSetMessages veröffentlichen, Werte aus der Host-Anwendung aktualisieren und WriterGroups/DataSets konfigurieren.
Subscriber	OPC UA PubSub Nachrichten empfangen, DataSetReader konfigurieren und empfangene Felder über High-Level-Events auswerten.
Transport	UDP Unicast, UDP Multicast, UDP Broadcast sowie MQTT-basierte PubSub-Kommunikation.
Encoding	UADP Binary Encoding und JSON Encoding entsprechend OPC UA PubSub Spezifikation.
Discovery	DataSetMetaData kann über PubSub Discovery bezogen werden; statisch konfigurierte Felder bleiben weiterhin möglich.
Security	MQTT über TLS, Zertifikatsprüfung über PKI-Struktur sowie überschreibbare Validierung durch die Host-Anwendung.

Adressierung per NodeId oder Browse-Pfad

NodeId	Klassische OPC UA Adressierung, z. B. ns=2;i=12345. Vollständig unterstützt und für fixe, bekannte Knoten geeignet.
Browse-Pfad	Lesbare, pfadbasierte Adressierung, z. B. Objects.Plant.Line1.Machine1.Temperature. Das SDK löst die passende NodeId im Hintergrund auf.
Praxisnutzen	Code bleibt verständlicher, wartbarer und weniger abhängig von serverinternen numerischen NodeIds. Gerade bei größeren Adressräumen beschleunigt dies Einarbeitung und Fehlersuche.

Historical Access und Datenspeicher

Historical Data und Historical Events sind sowohl für Client- als auch für Server-Szenarien vorgesehen. Für eigene Serveranwendungen stellt das SDK Schnittstellen bereit, über die projektspezifische Datenspeicher eingebunden werden können.

Client	Historische Werte lesen, z. B. Rohwerte, Werte zu bestimmten Zeitpunkten oder verarbeitete Werte; Historical Events lesen und überwachen.
Historical Update	Insert, Update, Replace und Delete historischer Werte, sofern der Zielserver diese Funktionen anbietet und zulässt.
Server	Eigene historische Werte und Events über einen OPC UA Historical Server bereitstellen.
Store-Interfaces	Austauschbare Schnittstellen für eigene History Stores und Event History Stores.
Mögliche Backends	Dateisystem, SQL-Datenbanken, Zeitreihendatenbanken, Cloud-Speicher, vorhandene Prozessdatenarchive oder projektspezifische Persistenzschichten.

Security, Zertifikate und CRA

Das SDK ist auf sichere Integration ausgelegt. Die konkrete Sicherheitskonfiguration bleibt bewusst unter Kontrolle des Integrators, weil Security Policies, Zertifikatsvertrauen, Benutzerverwaltung und Rechtekonzept immer zur Host-Anwendung und zum Anlagenumfeld passen müssen.

Bereich	Technischer Überblick
CRA	Die Komponente ist gemäß EU-Cyberresilienz-Verordnung 2024/2847 (CRA) zertifiziert. Technische Dokumentation, SBOM und Vulnerability-Scan-Prozess sind Teil des Security-Konzepts.
Security Modes	None, Sign, SignAndEncrypt. Für produktive Umgebungen wird SignAndEncrypt mit aktueller Security Policy empfohlen.
Security Policies	Aktuelle RSA-Policies wie Basic256Sha256, Aes128_Sha256_RsaOaep und Aes256_Sha256_RsaPss; .net zusätzlich ECC-Policies abhängig von Plattform/Build; deprecated Policies nur für Legacy-Kompatibilität.
UserTokenPolicies	Anonymous, UserName, Certificate und IssuedToken. In produktiven Umgebungen sollte Anonymous entfernt und UserName nur mit SignAndEncrypt verwendet werden.
PKI / Zertifikate	Automatische PKI-Struktur mit own, trusted, issuer und rejected; eigene Application-Zertifikate möglich; separates HTTPS/TLS-Zertifikat für opc.https.
Zertifikatsprüfung	Interne Prüfung u. a. auf Signatur, Gültigkeit, Trust, ApplicationUri, Hostname/Zertifikatskette; unbekannte Zertifikate werden in rejected abgelegt.
PubSub Security	Für MQTT-basierte PubSub-Kommunikation können TLS-Verbindungen mit Broker-Zertifikatsprüfung verwendet werden. Unbekannte Zertifikate werden in der PKI-Struktur abgelegt; eigene Validatoren können Trust-Entscheidungen bewusst überschreiben.
DoS-Schutz	Serverseitige Operation Limits und Session-Limits können eingesetzt bzw. angepasst werden, um übermäßige Requests zu begrenzen.

Hinweis: SecurityMode.None, Anonymous Access und Validatoren, die alle Zertifikate ungeprüft akzeptieren, sind für produktive Umgebungen nicht geeignet und sollten nur bewusst in isolierten Test- oder Sonderumgebungen verwendet werden.

Eigene Validatoren und Berechtigungssysteme

Zertifikatsvalidatoren	Schnittstellen, Events bzw. Listener zur Integration eigener Zertifikatsprüfungen und Trust-Entscheidungen.
Credential Validatoren	Serverseitige Anbindung eigener Benutzer-/Passwortprüfungen, z. B. gegen eine eigene Benutzerverwaltung, Datenbank oder vorhandene Identity-Infrastruktur.
Permission Validatoren	Integration eigener Rechteprüfungen für Rollen, Gruppen, einzelne Benutzer oder projektspezifische Berechtigungsmodelle.
Feingranularität	Berechtigungen können projektabhängig sehr fein umgesetzt werden, theoretisch bis auf Ebene einzelner Knoten im OPC UA Adressraum.
Typische Zielsysteme	Eigene Benutzerverwaltung, Rollen-/Gruppenmodell, SQL-/Datei-basierte Rechteverwaltung, LDAP/Active Directory, SSO-/Token-Verfahren oder kundenspezifische Authentifizierungslogik - sofern durch die Host-Anwendung angebunden.

Performance- und Komfortfunktionen

Schnelle Einarbeitung	Viele typische OPC UA Aufgaben sind über High-Level-Funktionen und Workshops mit wenigen Codezeilen nachvollziehbar.
Lesbare Adressierung	Browse-Pfade ergänzen klassische NodeIds und machen Code sowie Konfigurationen verständlicher.
Batch-Zugriffe	Mehrere Lese- und Schreiboperationen können in einem Aufruf gebündelt werden.
Registered Read/Write	Performanter Zugriff auf häufig verwendete Knoten durch Registrierung beim Server, sofern passend zum Szenario.
Verbindungsmanagement	Automatisches Connect/Reconnect/Disconnect, Keep-Alive-Monitoring und klare Events bei Verbindungsstatus und Fehlerfällen.
Serverbetrieb	Operation Limits, konfigurierbare Security Policies, Logging und Diagnoseereignisse unterstützen robuste Serveranwendungen.

Weitere Funktionen im Überblick

Kategorie	Funktionen
Client	Discovery, Connect, Read/Write, Browse, Browse by Path, Attribute lesen, Subscriptions, Monitored Items, Method Calls, Alarme/Events, Historical Access, Reverse Connect.
Server	Eigener Adressraum, Variablen, Arrays, Properties, Methoden, dynamische Knoten, mehrere Namespaces, Custom Types, NodeSet2 Import, Logging, Reverse Connect.
Historical	Historical Data, Historical Events, Historical Update, eigene History Stores, eigene Event History Stores, Anbindung vorhandener Persistenzsysteme.
PubSub	Publisher, Subscriber, UADP, JSON, UDP Unicast/Multicast/Broadcast, MQTT, TLS, PubSub Discovery und DataSetMetaData.
Security	Security Policies, UserTokenPolicies, PKI, Zertifikatsprüfung, eigene Zertifikatsvalidatoren, eigene Credential-/Permission-Validatoren, Operation Limits.
Integration	NuGet/Maven/Auslieferungspaket, optionales PubSub Add-on, direkte Einbindung in Host-Anwendungen, keine DCOM-Infrastruktur, Client/Server/PubSub im selben Projekt möglich.
Einarbeitung	Ausführliche Workshops und Beispielprojekte für .net und Java, abgestimmt auf typische Client-, Server- und PubSub-Aufgaben.

Professioneller Support, Wartungspläne (Maintenance Plan)

Für PLCcom Opc Ua Sdk sind Wartungspläne mit einer Laufzeit von 1, 2 oder 3 Jahren verfügbar. Während der Laufzeit werden alle Upgrades und Updates kostenlos bereitgestellt, außerdem ist technischer Support bei technischen Problemen enthalten.

Nach Ablauf des Wartungsplans erfolgt keine automatische Verlängerung. Auf Wunsch kann der Wartungsplan verlängert werden - vor Ablauf erhalten Sie dazu ein Angebot.

Unabhängig davon sind sicherheitsrelevante Updates im Rahmen des für die jeweilige Major-Version definierten Security-Support-Zeitraums vorgesehen.

Hinweis:

Alle Produktnamen oder andere Namen oder Marken, auf die in diesem Dokument Bezug genommen wird, sind Warenzeichen oder eingetragene Warenzeichen und Eigentum ihrer jeweiligen Inhaber. Es besteht keinerlei Verbindung zwischen der genannten Marke oder dem Markeninhaber und der Fa. Indi.An GmbH. Jegliche Nennung von Marken dient ausschließlich als Hinweis zur Nutzung und Verwendungszweck.