What is PLCcom Opc Ua Sdk?

PLCcom Opc Ua Sdk provides .NET and Java developers with a powerful foundation for integrating OPC UA client and server functionality quickly and reliably into their own applications.

Typical OPC UA tasks such as connection setup, reading, writing, subscriptions, events, methods or providing custom server address spaces are already encapsulated as convenient functions.
This allows even demanding OPC UA solutions to be implemented cleanly and efficiently **without deep OPC UA specialist knowledge**.
The SDK supports the current **OPC UA Specification 1.05** and is designed for fast, robust integration.In addition to classic NodeId addressing, the SDK optionally supports access via **readable browse paths** such as **Objects.Plant.Machine1.Temperature**. This makes applications easier to understand and maintain without having to rely exclusively on technical NodeIds such as ns=2;i=1001.

For professional client and server applications, PLCcom Opc Ua Sdk provides flexible interfaces for custom certificate validation, custom user and permission concepts down to node level and custom data stores for Historical Data and Historical Events.
This makes the SDK a flexible foundation for individual OPC UA solutions – from HMI/SCADA and machine applications to gateways, diagnostic tools, test benches and industry-specific automation software.

The component is certified in accordance with Regulation (EU) 2024/2847, the Cyber Resilience Act (CRA).

Technical overview

Publisher	Indi.An GmbH (Copyright ©)
System	SDK/library for OPC UA client, server and PubSub communication
Current version	10.x
Environment	.NET and Java, depending on the delivery package
OPC UA specification	aligned with OPC UA Specification 1.05
Operating modes	Client SDK, Server SDK, PubSub add-on or combined operation within the same host application
Transport	OPC UA Binary over TCP (opc.tcp), OPC UA over https (opc.https / https) and OPC UA PubSub over UDP and MQTT. http is technically supported but not recommended for production environments.
Security	SecurityMode Sign / SignAndEncrypt, modern RSA policies; .NET additionally supports ECC policies depending on platform/build. PubSub supports TLS-based MQTT connections and PKI-based certificate validation.
CRA	Certified in accordance with **Regulation (EU) 2024/2847, the Cyber Resilience Act (CRA)**. Technical documentation, SBOM and vulnerability scan processes are part of the security concept.

Code examples and workshops

The example repositories are not just short code snippets. They are structured as coherent workshops and guide developers step by step through typical client and server tasks. This makes them especially useful for onboarding new developers.

Area	Content
.NET repository	C# and VB workshops for client, server and PubSub scenarios, including Data Access, alarms, Historical Access, PubSub over UDP/MQTT, NodeSet import, logging and Reverse Connect.
Java repository	Java workshops for client, server and PubSub scenarios, including Data Access, Alarms & Conditions, Historical Data/Events, PubSub over UDP/MQTT, complex data types, NodeSet2 import, logging and Reverse Connect.
Online help	API documentation, class reference and detailed technical information for the respective .NET or Java version.
Delivery package	Contains additional help files, examples and documentation matching the respective delivery package.

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

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

Licensing by license key

A license key is required for operation (user/serial). Trial keys are available.

Note: You can generate a 14-day trial key on our website: https://www.indi-an.com/en/plccom/opc-ua-sdk/opcua-download/

System requirements

PLCcom Opc Ua Sdk is a pure .NET or Java library, depending on the delivery package. There are no special hardware requirements; the requirements of the host application, selected runtime and target environment are decisive.

.NET Framework	.NET Framework 4.7.2 and 4.8 or later (.NET Framework packages).
.NET / .NET Standard	.NET 5, .NET 6 and .NET 7 are covered via .NET Standard 2.1. Direct packages are additionally available for .NET 8, .NET 9 and .NET 10.
.NET integration	Integration as NuGet package PLCcom.Opc.Ua.Sdk; the optional PubSub add-on is provided version-aligned with the SDK. Download: https://www.indi-an.com/en/plccom/opc-ua-sdk/opcua-download/
.NET development	Visual Studio 2022 or later is recommended for the example projects.
Java runtime	Java JDK/JRE/OpenJDK 8 or later; tested up to Java 26.
Java integration	Integration as Maven artefact com.indi-an.plccom:plccom-opc-ua-sdk; the optional PubSub add-on com.indi-an.plccom:plccom-opc-ua-pubsub is provided version-aligned with the SDK. Download: https://www.indi-an.com/en/plccom/opc-ua-sdk/opcua-download/
Java development	Maven and a current Java IDE such as IntelliJ IDEA, Eclipse, NetBeans or Visual Studio Code with Java extensions are recommended.

Operating systems	.NET Framework variants are typically Windows-oriented. Modern .NET and Java variants can be used cross-platform according to the respective runtime.

Deployment / integration

Direct integration	The SDK is integrated directly into the host application. The application controls lifecycle, connection setup, security configuration and data model.
.NET delivery	a. NuGet package <PackageReference Include="PLCcom.Opc.Ua.Sdk" Version="10.*" /> b. optional PubSub add-on, version-aligned with the SDK c. optional delivery package: https://www.indi-an.com/en/plccom/opc-ua-sdk/opcua-download/
Java delivery	a. Maven artefact com.indi-an.plccom:plccom-opc-ua-sdk b. optional Maven artefact com.indi-an.plccom:plccom-opc-ua-pubsub, version-aligned with the SDK c. optional delivery package: https://www.indi-an.com/en/plccom/opc-ua-sdk/opcua-download/
No OPC Classic legacy stack	No COM registration, no DCOM configuration and no separate OPC Classic infrastructure required.
Client / Server / PubSub	Client, server and PubSub functionality can be used separately or together within one application.
Not standalone	PLCcom Opc Ua Sdk is a library. It is not a standalone OPC UA server/client, but an integration component for custom software.

Architecture and operating modes

Client mode	Host application -> PLCcom Opc Ua Sdk -> OPC UA Server. Typical for HMI/SCADA, test benches, gateways, diagnostic and analysis software.
Server mode	OPC UA Client -> PLCcom Opc Ua Sdk -> host application / data model. Typical for custom machine, line, gateway or simulation servers.
Combined mode	Client and server functionality running in parallel in the same application, e.g. as a gateway between existing OPC UA servers and a custom, unified address space.
Address space	Server-side objects, variables, arrays, properties, methods, alarms, events, multiple namespaces and imported NodeSet2 models can be provided.

Functions OPC UA Client SDK

The Client SDK provides convenient high-level functions so typical OPC UA tasks can be implemented quickly without manually exposing the full complexity of the OPC UA stack.

Topic	Technical scope
Discovery / connection	Discover servers and endpoints, select endpoint, create session, use user or certificate authentication.
Data Access	Read and write values by NodeId or browse path, batch operations, read/write attributes.
Subscriptions	Monitor variables through subscriptions and monitored items, including data change and event scenarios.
Methods	Call OPC UA methods, including structured input and output parameters.
Alarms / Events	Subscribe to alarms and events, build alarm lists, process Acknowledge, Confirm and comments.
Historical Access	Read historical data, retrieve values at specific timestamps, use raw and processed values, read/monitor historical events.
Performance	Batch access, Registered Read/Write for frequently used nodes, keep-alive monitoring and clear connection events.
Reverse Connect	Connections in restricted network architectures, e.g. where classic inbound connections are difficult due to firewalls.

Functions OPC UA Server SDK

With the Server SDK, a custom .NET or Java application can expose an OPC UA server. The host application provides the data model, values, security decisions and, if required, historical persistence.

Topic	Technical scope
Address space	Folders, objects, variables, arrays, properties, methods, multiple namespaces and dynamic nodes at runtime.
Data Access Server	Provide custom variables and values, update them and make them readable/writable for OPC UA clients.
Methods	Publish custom server methods with callback logic and allow OPC UA clients to call them.
Types / models	Custom structured data types, complex data types, custom ObjectTypes/VariableTypes and NodeSet2 XML import.
Alarms & Conditions	Alarm models with states, limit/discrete/dialog alarms, Acknowledge, Confirm, comments and event output.
Historical Server	Provide historical values and historical events; connect persistence through exchangeable store interfaces.
Security / users	UserTokenPolicies, user/password validation, certificate authentication, roles and custom credential/permission validators.
Diagnostics / operation	Logging, connection events, operation limits, session limits and configurable security policies.

OPC UA PubSub Add-on

The optional PubSub add-on extends the SDK with OPC UA PubSub publisher and subscriber functionality. It builds on the existing OPC UA stack and provides high-level configurations for typical PubSub scenarios.

Topic	Technical scope
Publisher	Publish OPC UA PubSub DataSetMessages, update values from the host application and configure WriterGroups/DataSets.
Subscriber	Receive OPC UA PubSub messages, configure DataSetReaders and process received fields through high-level events.
Transport	UDP unicast, UDP multicast, UDP broadcast and MQTT-based PubSub communication.
Encoding	UADP binary encoding and JSON encoding according to the OPC UA PubSub specification.
Discovery	DataSetMetaData can be obtained through PubSub Discovery; statically configured fields remain possible.
Security	MQTT over TLS, certificate validation through the PKI structure and validation that can be deliberately overridden by the host application.

Addressing by NodeId or browse path

NodeId	Classic OPC UA addressing, e.g. ns=2;i=12345. Fully supported and suitable for fixed, known nodes.
Browse path	Readable path-based addressing, e.g. Objects.Plant.Line1.Machine1.Temperature. The SDK resolves the matching NodeId in the background.
Practical benefit	Code remains more understandable, maintainable and less dependent on server-internal numeric NodeIds. Especially in larger address spaces, this accelerates onboarding and troubleshooting.

Historical Access and data stores

Historical Data and Historical Events are intended for both client and server scenarios. For custom server applications, the SDK provides interfaces through which project-specific data stores can be integrated.

Client	Read historical values, e.g. raw values, values at specific timestamps or processed values; read and monitor Historical Events.
Historical Update	Insert, Update, Replace and Delete historical values, provided the target server offers and permits these functions.
Server	Expose custom historical values and events through an OPC UA Historical Server.
Store interfaces	Exchangeable interfaces for custom History Stores and Event History Stores.

Possible backends	File system, SQL databases, time-series databases, cloud storage, existing process data archives or project-specific persistence layers.

Security, certificates and CRA

The SDK is designed for secure integration. The specific security configuration intentionally remains under the control of the integrator because security policies, certificate trust, user management and authorization concepts must always match the host application and plant environment.

Area	Technical overview
CRA	Certified in accordance with Regulation (EU) 2024/2847, the Cyber Resilience Act (CRA). Technical documentation, SBOM and vulnerability scan processes are part of the security concept.
Security Modes	None, Sign, SignAndEncrypt. For production environments, SignAndEncrypt with a current security policy is recommended.
Security Policies	Current RSA policies such as Basic256Sha256, Aes128_Sha256_RsaOaep and Aes256_Sha256_RsaPss; .NET additionally supports ECC policies depending on platform/build; deprecated policies only for legacy compatibility.
UserTokenPolicies	Anonymous, UserName, Certificate and IssuedToken. In production environments, Anonymous should be removed and UserName should only be used with SignAndEncrypt.
PKI / certificates	Automatic PKI structure with own, trusted, issuer and rejected; custom application certificates possible; separate HTTPS/TLS certificate for opc.https.
Certificate validation	Internal checks include signature, validity, trust, ApplicationUri, hostname/certificate chain; unknown certificates are stored in rejected.
PubSub security	For MQTT-based PubSub communication, TLS connections with broker certificate validation can be used. Unknown certificates are stored in the PKI structure; custom validators can deliberately override trust decisions.
DoS protection	Server-side operation limits and session limits can be used or adjusted to restrict excessive requests.

Note: SecurityMode.None, anonymous access and validators that accept all certificates without validation are not suitable for production environments and should only be used deliberately in isolated test or special environments.

Custom validators and authorization systems

Certificate validators	Interfaces, events or listeners for integrating custom certificate validation and trust decisions.
Credential validators	Server-side integration of custom user/password checks, e.g. against a custom user administration, database or existing identity infrastructure.
Permission validators	Integration of custom authorization checks for roles, groups, individual users or project-specific permission models.
Granularity	Permissions can be implemented with very fine granularity depending on the project, theoretically down to individual nodes in the OPC UA address space.
Typical target systems	Custom user administration, role/group models, SQL- or file-based permission management, LDAP/Active Directory, SSO/token-based procedures or customer-specific authentication logic - if connected by the host application.

Performance and convenience functions

Fast onboarding	Many typical OPC UA tasks can be understood through high-level functions and workshops with just a few lines of code.
Readable addressing	Browse paths complement classic NodeIds and make code and configurations easier to understand.
Batch access	Multiple read and write operations can be bundled in one call.
Registered Read/Write	High-performance access to frequently used nodes by registering them with the server, where suitable for the scenario.
Connection management	Automatic connect/reconnect/disconnect, keep-alive monitoring and clear events for connection status and error cases.
Server operation	Operation limits, configurable security policies, logging and diagnostic events support robust server applications.

Further functions at a glance

Category	Functions
Client	Discovery, Connect, Read/Write, Browse, Browse by Path, read attributes, subscriptions, monitored items, method calls, alarms/events, Historical Access, Reverse Connect.
Server	Custom address space, variables, arrays, properties, methods, dynamic nodes, multiple namespaces, custom types, NodeSet2 import, logging, Reverse Connect.
Historical	Historical Data, Historical Events, Historical Update, custom History Stores, custom Event History Stores, integration of existing persistence systems.
PubSub	Publisher, subscriber, UADP, JSON, UDP unicast/multicast/broadcast, MQTT, TLS, PubSub Discovery and DataSetMetaData.
Security	Security Policies, UserTokenPolicies, PKI, certificate validation, custom certificate validators, custom credential/permission validators, operation limits.
Integration	NuGet/Maven/delivery package, optional PubSub add-on, direct integration into host applications, no DCOM infrastructure, Client/Server/PubSub possible in the same project.
Onboarding	Extensive workshops and example projects for .NET and Java, aligned with typical client, server and PubSub tasks.

Professional support, maintenance plans

Maintenance plans with a term of 1, 2 or 3 years are available for PLCcom Opc Ua Sdk. During the term, all upgrades and updates are provided free of charge; technical support for technical issues is also included.

After the maintenance plan expires, there is no automatic renewal. The maintenance plan can be extended on request - you will receive an offer before it expires.

Security-relevant updates are independently planned within the security support period defined for the respective major version.

Note:

All product names or other names or trademarks referenced in this document are trademarks or registered trademarks and are the property of their respective owners. There is no affiliation between the named trademark or trademark owner and Indi.An GmbH. Any mention of trademarks is solely intended as a reference to use and intended purpose.