OPC UA SDK
Client + Server Development

Versionshistorie

Getestet? Du willst es?

Lizenzmodell Preise Angebot Jetzt bestellen

Historie

2.41.1.0 (freigegeben am 18.01.2024)

VERÄNDERT

- Die Microsoft.AspNetCore.Server.Kestrel.Core-Abhängigkeit wurde auf Version 2.1.25 aktualisiert, um eine Sicherheitslücke zu beheben.

BEHOBEN

- Problem, das ein Speicherleck im OpcClient verursachte, wenn OpcClient.BrowseNodes() wiederholt aufgerufen wurde oder auf Dateiknoten mit der Klasse OpcFileMethods, OpcFileStream oder OpcFileInfo zugegriffen wurde.
- Problem, das dazu führen konnte, dass der OpcClient fälschlicherweise Leseanfragen (vom Keep-alive-Timer) sendete, selbst wenn der ActivateSession-Aufruf noch nicht abgeschlossen war.

2.41.0.0 (freigegeben am 02.12.2023)

VERÄNDERT

- Die Benutzerauthentifizierung im OpcServer wird jetzt nur noch einmal während ActivateSession durchgeführt, anstatt für jede Anfrage einzeln.

BEHOBEN

- Problem mit OpcVariableNodeInfo.IsArray und OpcVariableNode.IsArray, die bei einem Array mit 3 oder mehr Dimensionen fälschlicherweise false zurückgeben (ersteres seit v2.33.0).
- Problem mit einem Deadlock, das beim Aufruf von OpcClient.Disconnect() oder OpcClient.Dispose() auftrat, während gleichzeitig Ereignisabonnements aktiv waren.
- Problem, das dazu führte, dass der OpcClient.State nicht auf 'Reconnecting' wechselte, wenn eine Verbindungsunterbrechung erkannt wurde, während eine Operation wie ReadNodes() noch ausgeführt wurde.
- Problem, das die Erstellung eines Ereignisfilters aus einer OpcNodeId verhinderte, wenn ein aufgelöster Namespace verwendet wurde.
- Problem, das ein Speicherleck verursachte, wenn wiederholt OpcMethodNode-Instanzen erstellt (und verworfen) wurden, wenn eine Delegate-Instanz übergeben wurde.
- Problem, das dazu führte, dass reflektierte Typfelder in einer falschen Reihenfolge deklariert wurden, wenn der Typ eine Schnittstelle implementiert.

2.40.0.0 (freigegeben am 23.10.2023)

NEU

- Einführung der neuen OpcContext.Enter(...)-Methoden zum Eintritt in einen lokalen Kontext, um Nicht-Client/Server-Szenarien durchzuführen.
- OpcDataTypeInfo wurde erweitert, um ExtensionObjects auch als Systemtyp zu unterstützen (IsSystemType-Eigenschaft).
- Einführung neuer OpcData.Decode<T>(...) Methoden zur manuellen Dekodierung eines binär kodierten Datenstroms/Puffers in ein strukturiertes Objekt.
- Einführung einer expliziten Unterstützung für die Kodierung/Dekodierung/Verwendung von ExtensionObjects.
- Verbesserte Leistung beim erstmaligen Empfang von benutzerdefinierten Ereignisdaten.

- Einführung der neuen Methode OpcNodeSet.Load(string, OpcNodeSetLoadOptions), um mehr Details über die zu ladende NodeSet-Datei bereitzustellen.

BEHOBEN

- Problem mit falsch berechneter Kodierungsmaske bei Verwendung expliziter Kodierung.
- Problem mit der Kodierung/Dekodierung von Subtypen eingebauter Datentypen entsprechend ihren Basistypen.
- Problem mit nicht gemeldeten Ereignissen am „Server“-Knoten, solange kein überwachtes Element erstellt wurde, um das erwartete Ereignis in seiner Knotenhierarchie zu überwachen, in der es sich befindet.
- Fehlender impliziter Aufruf von ApplyChanges mit OpcEventNode.ReportEvent(…).
- Probleme mit der En-/Decodierung von ExtensionObjects in benutzerdefinierten strukturierten Datentypen.
- Problem mit verschachtelten Ereignistyp-Feldern wie ActiveState/Id, die seit v2.32.0 nicht korrekt gemeldet werden.
- Problem mit gemischten Datentypen, die mit DataTypeDescriptions und DataTypeDefinitions definiert werden, während die Definitionen unterschiedliche Feldtypen verwenden. Die Beschreibungen werden jetzt priorisiert.
- Problem mit Ereignisabonnements, die seit v2.29.0 nicht funktionieren, wenn kein OpcEventFilter angegeben wurde.

2.33.0.0 (freigegeben am 11.09.2023)

NEU

- OpcNodeContext wurde um die neue Eigenschaft NodeManager erweitert, die auf den OpcNodeManager im aktuellen Kontext verweist, in dem eine Operation ausgeführt wird.
- Einführung der Methode OpcMethodNode.Call(…) zum Aufrufen eines implementierten Methodenknotens unter Verwendung eines benutzerdefinierten Kontextes unabhängig von einem „Client-Aufruf“.
- Einführung der Methoden Close(…), GetPosition(…), Open(…), Read(…), SetPosition(…) und Write(…) zum Aufrufen der implementierten Methoden eines OpcFileNode unter Verwendung eines benutzerdefinierten Kontexts unabhängig von „Client-Aufrufen“.
- Verbesserte Leistung bei der Erstellung von Ereignisfiltern, die nur Ereignistypen des vom Server bereitgestellten Knotentypsystems verwenden.
- Unterstützung von Datentypen, die nur über eine Datentypdefinition und nicht mehr über eine Datentypbeschreibung deklariert werden.
- Unterstützung der dynamischen Auflösung und Darstellung von Union-Datentypen unter Verwendung des OpcDataObjects.

VERÄNDERT

- OpcContext.Empty.SessionId bietet nicht mehr eine Null-Referenz (Nothing in Visual Basic). Stattdessen liefert es OpcNodeId.Null.
- Die Zugriffsmodifikatoren der Methoden OpcNodeManager.AddNode(…) und OpcNodeManager.RemoveNode(…) wurden von protected auf public geändert, um die Benutzerfreundlichkeit zu verbessern.
- Auswertung von OpcVariableNodeInfo.IsArray verwendet nun den ValueRank-Wert anstelle der ArrayDimensions, um zu entscheiden, ob das Value-Attribut ein Array liefert.

2.32.0.0 (freigegeben am 28.08.2023)

NEU

- Weitere DCOM-Sicherheitsoptionen mit Eigenschaften zur Konfiguration von AuthenticationService, AuthorizationService und ProxyCapabilities aus dem Anwendercode.
- Einführung der neuen Eigenschaft Name in OpcDataTypeAttribute, um für jeden DataType einen anderen Namen zu definieren, der vom implementierenden .NET-Typ verwendet wird.
- Einführung der Unterstützung für die verzögerte Initialisierung von benutzerdefinierten OpcData Store-Implementierungen mit Unterstützung für einen Standardkonstruktor.
- Einführung der neuen Eigenschaft IsUnion in IOpcDataTypeInfo, um anzugeben, ob ein Datentyp ein Union-Datentyp ist.
- Einführung der neuen Schnittstelle IOpcUnion zur Implementierung von benutzerdefinierten Union-Datentypen unter Verwendung dieses Schnittstellentyps.
- Einführung einer typisierten Unterstützung für den abstrakten Datentyp Number, der die Kodierung aller primitiven numerischen Datentypen unterstützt.
- Einführung der neuen Klasse OpcSubscriptionSetup für die Erstellung eines neuen Abonnements mit allen vordefinierten Eigenschaften.
- Einführung der neuen Methode OpcClient.CreateSubscription(...), die ein OpcSubscriptionSetup verwendet, um eine neue OpcSubscription zu erstellen.

BEHOBEN

- Es wurde ein Verweis auf System.Text.Encodings.Web 4.5.1 (für das NuGet-Paket) hinzugefügt, um eine Warnung vor einer Sicherheitslücke zu beheben, die durch den transitiven Verweis über Microsoft.AspNetCore.Server.Kestrel 2.1.3/1.1.3 verursacht wurde.
- Problem mit Event, die nicht auf dem „Server“-Knoten gemeldet werden, bis ein MonitoredItem erstellt wurde, um das erwartete Event in seiner Knotenhierarchie zu überwachen, wo es sich befindet.
- Fehlender impliziter Aufruf von ApplyChanges mit OpcEventNode.ReportEvent(...).
- Problem mit der Verwendung des BaseEventType-Typbezeichners in den einfachen Attributoperanden in einem EventFilter.
- Problem, um sicherzustellen, dass gleiche Start- und Endzeiten beim Lesen von verarbeiteten History-Werten nicht zu einer Endlosschleife und steigenden Speicherverbrauch führen.
- Probleme mit Deadlocks bei der Verwendung von UseDynamic und Subscriptions, die strukturierte Datentypen verwenden, die mit OpcDataObjects dargestellt werden.

2.31.0.0 (freigegeben am 17.05.2023)

VERÄNDERT

- Aktualisierte Newtonsoft.Json Abhängigkeit auf Version 13.0.3, um eine Sicherheitslücke zu beheben.
- Der UA Wrapper Server für OPC Classic Verbindungen generiert nun immer ein neues Anwendungszertifikat, anstatt zu versuchen, eines aus dem Zertifikatspeicher des Clients zu laden.
- Wenn bei einer „opc.com:“-URL kein Port angegeben ist, wird der Port für den Classic Wrapper Server nicht mehr von der classID und progID abgeleitet, sondern es wird eine freie dynamische Portnummer ermittelt. Dies erleichtert die gleichzeitige Verwendung mehrerer OpcClients mit der gleichen „opc.com:“-URL.

BEHOBEN

- Problem bei der Aktualisierung der ApplicationUri des OpcServers aus dem ausgewählten Zertifikat, das nur für den zweiten Start statt für den ersten Start angewendet wurde, was dazu führte, dass

der Reconnect-Mechanismus des OpcClient fehlschlug.

2.30.0.0 (freigegeben am 07.12.2022)

NEU

- Trace-Protokollierung auf Debug-Ebene bei Verwendung der Unterbrechungsüberwachung mit dem OPC Classic UA Wrapper Server hinzugefügt.
- Die Eigenschaft OpcNodeId.Resolution.InheritParentNamespace wurde eingeführt, um die Vererbung des Namespace von Elternknoten an ihre Kinder zu steuern.
- Die Auflösung des Standard-Node-Identifiers verwendet nun den Namespace (wenn kein Standardnamespace) des Elternknotens, um den „automatischen“ Node-Identifier eines Knotens zu definieren (falls OpcNodeId.Resolution.InheritParentNamespace gleich true ist; der Standardwert).

VERÄNDERT

- Das Verhalten beim Starten und Beenden eines OPC Classic UA Wrapper Servers ist nun deterministischer, da jeder OpcClient einen dedizierten Wrapper Server verwendet, der nun auch gestoppt wird, wenn der OpcClient die Verbindung trennt.
- Die Methode OpcCertificateIdentity.Matches() ist jetzt virtuell, um einen benutzerdefinierten Identitätsvergleich zu ermöglichen.
- Die Eigenschaft OpcEvent.EventId liefert nun den zugrundeliegenden ByteString anstelle eines Byte-Arrays (bevor der ByteString eingeführt wurde).

BEHOBEN

- Problem mit der ArgumentOutOfRangeException, die ausgelöst wird, wenn ein OpcFiniteStateMachineNode ohne Definition von Zuständen über Attribute erstellt wird.
- Problem mit NullReferenceException, wenn ein OpcFiniteStateMachineNode erstellt wird, nachdem der OpcServer bereits gestartet wurde.
- OpcEvent.GetValue<T>(string) gibt korrekt einen ByteString-Wert zurück, wenn der Typparameter T Opc.UaFx.ByteString ist.
- Problem mit ArgumentException in OpcClient.Connect(), wenn der Friendly Name der aktuellen AppDomain Nicht-Pfad-Zeichen enthält (z.B. in einem Office VSTO Add-on).

2.29.0.0 (freigegeben am 16.09.2022)

NEU

- Eigenschaft OpcClientInterfaces.UseBreakMonitoring hinzugefügt, um den UA Wrapper Server für OPC Classic Verbindungen vorübergehend anzuhalten, während ein (DCOM) Verbindungsabbruch zum OPC Classic Server erkannt wird.
- Einführung der Methode OpcNamespace.GetNextId() zur Abfrage des nächsten freien numerischen OpcNodeId aus dem OpcNamespace.
- Einführung einer Reihe von neuen Attributen für spezifische OpcStateMachineNode-Zustände und deren Übergänge.
- Einführung der neuen OpcStateNode Klasse zur Darstellung des OPC UA StateType.
- Einführung der neuen OpcTransitionNode Klasse zur Darstellung des OPC UA TransitionType.
- Unterstützung für die Konfiguration von benutzerdefinierten OpcFiniteStateMachineNodes unter Verwendung der neuen Zustands-/Übergangsattribute, um den Anfangszustand, die Zustandsknoten und die Übergangsknoten unter Verwendung von numerischen oder Enum-Werten einzustellen.
- Einführung der Methode OpcFiniteStateMachineNode.ChangeStateTo(), um den aktuellen Zustand der StateMachine in einen anderen Zustand zu ändern, einschließlich der Validierung seiner Existenz

und Übersetzbarkeit.

VERÄNDERT

- OpcNode.AddNotifier wird bei anderen Typen als OpcObjectNodes und seinen Unterklassen nicht mehr unterstützt.

BEHOBEN

- Probleme mit fehlenden Datentypinformationen für den abstrakten Datentyp „Structure“ (i=22).
- Probleme beim Zugriff auf die Eigenschaften OpcTransitionVariableNode.VariableId/Node.
- Probleme beim Zugriff auf die Eigenschaften von OpcFiniteTransitionVariableNode.VariableId/Node.
- Probleme beim Zugriff auf die Eigenschaften von OpcStateVariableNode.VariableId/Node.
- Probleme beim Zugriff auf die Eigenschaften von OpcFiniteStateVariableNode.VariableId/Node.

2.28.1.0 (freigegeben am 02.09.2022)

NEU

- Einführung der Inline-Initialisierung von „reservierten“ Feldern im Zusammenhang mit Kodierungsmasken unter Verwendung eines Bit-Arrays.

BEHOBEN

- Problem mit Hostname/IP Adresse, die bei der Generierung des Ports für den OPC Classic UA Wrapper Server nicht berücksichtigt wird.
- Problem mit OPC Classic Client, der immer noch IOPCBrowseServerAddressSpace.ChangeBrowsePosition() mit OPC_BROWSE_TO aufruft, obwohl OpcClient.Interfaces.DataAccess.SupportsBrowseTo auf false gesetzt wurde.
- Problem mit falschem ValueRank und ArrayDimensions in Datentypdefinitionen von Strukturen, die ein anderes Feld zur Definition der Länge eines Array-Feldes verwenden.

2.28.0.0 (freigegeben am 19.08.2022)

NEU

- Die Inline-Definition von DataTypeDictionaryNodes wurde dahingehend verbessert, dass nur noch einer zur Verfügung steht, wenn eine Datentypbeschreibung eines Datentyps veröffentlicht werden soll.
- Unbekannte (benutzerdefinierte) Datentypen werden jetzt mit den Ereignissen OpcData.TypeResolve und OpcClient.TypeResolve unter Verwendung der entsprechenden Kodierungskennung und OpcNodeId.Null für die Datentypkennung aufgelöst.

VERÄNDERT

- Die Deaktivierung von OpcAutomatism.UseDynamicTypeRegistration führte zu einer verzögerten dynamischen Typregistrierung, wenn ein benutzerdefinierter Datentyp manuell registriert wurde.

BEHOBEN

- Problem mit ArgumentException, wenn die Eigenschaft Value eines OpcStateVariableNode oder OpcFiniteStateVariableNode geändert wurde.
- Problem mit OpcData.TypeResolve, bei dem die übergebenen Ereignisargumente den Encoding Identifier als Typbezeichner lieferten. Dies wurde behoben und der Encoding Identifier wird als Teil des Encodings übergeben.

- Problem, dass OpcClient.TypeResolve nicht für jeden Datentyp aufgerufen wird.
- Problem mit nicht kodierten dynamisch aufgelösten Datentypen.
- Problem mit fehlender führender Array-Länge, wenn ein Datentyp ein Feld mit fester Länge verwendet.
- Probleme mit inline registrierten Datentypen, obwohl OpcAutomatism.UseDynamicTypeRegistration deaktiviert wurde.
- Probleme mit Aufzählungen, die in einem anderen DataTypeDictionary als strukturierte Datentypen beschrieben werden, wenn voll qualifizierte Datentypbezeichner (Wert und Namespace Uri) verwendet werden.
- Problem beim Importieren von Node-Sets mit Struktur-Datentyp-Definitionen ohne explizite Basistyp-Kennung.
- Problem mit Fehlern beim Laden von Datentypen in einigen Clients von Drittanbietern beim Zugriff auf DataTypeDictionaries, die nur Aufzählungen deklarieren.
- Probleme bei der Auswahl eines Endpunkts durch Constraints, mit Endpunkte die Mode == None aber Algorithm != None haben und Fallback, wenn kein anderer Endpunkt verfügbar ist.

2.27.0.0 (freigegeben am 27.06.2022)

NEU

- Neue Eigenschaft NumberInList zu OpcOrderedObjectNode hinzugefügt, um die aktuelle Position eines Knotens innerhalb des Servers zu bestimmen.
- Neues Ereignis OpcClient.TypeResolve hinzugefügt, um benutzerdefinierte Typen gemäß Client-/sitzungsbezogenen Bedingungen aufzulösen.
- OpcDataTypeDictionary.GetType(OpcEncoding) wurde verbessert, um Kodierungen ohne erweiterte Knotenbezeichner zu berücksichtigen.
- Einführung neuer geschützter statischer (Shared in Visual Basic) OpcNode.DefineModel-Methoden zur Definition von knotentypabhängigen Modellierungsregeln für das gesamte Node-Modell.
- Einführung von IOpcDataTypeMapper und seiner Standardimplementierung OpcDataTypeMapper.
- Einführung einer neuen OpcData-Methode zur Registrierung, Bestimmung und Aufhebung der Registrierung von benutzerdefinierten Datentyp-Mappern.
- Einführung der neuen Methoden OpcDataTypeMapper.Convertible und OpcDataType.Delegate, um einfache benutzerdefinierte Datentyp-Mapper zu erstellen.

BEHOBEN

- Problem mit OpcPropertyNode<T>, das nicht zu seinem Eltern-Knoten hinzugefügt wird, wenn der Typ-Parameter einen benutzerdefinierten Datentyp darstellt.
- Problem mit BadNodeIdUnknown bei EnumStrings, wenn ein benutzerdefinierter Aufzählungstyp mit EnumStrings definiert ist.
- Problem mit mehrfachen hierarchischen Referenzen zwischen Eltern- und Kindknoten, wenn ein Subtyp des Standardreferenztyps für einen Kindknoten verwendet wird.
- Regression mit Erweiterung der IOpcNodeInfo.Child Überladungen und deren Implementierungen.

2.26.0.1 (freigegeben am 13.04.2022)

BEHOBEN

- Regression, die die Unterklassifizierung von benutzerdefinierten Datentypen ohne Datentypattribute deaktiviert.

2.26.0.0 (freigegeben am 13.04.2022)

NEU

- Das OpcDataTypeAttribute ermöglicht jetzt die Definition, welche Datentypen mit Hilfe von Datentypdefinitionen und/oder Datentypbeschreibungen deklariert werden sollen.
- Neue Declarations-Eigenschaft zu OpcDataTypeInfo hinzugefügt, um anzuzeigen, auf welche Weise der Datentyp deklariert wurde.
- Unterstützung für Arrays von Bit-Membern zur Definition der Kodierungsbits hinzugefügt, so dass nicht mehr nur skalare Kodierungsbit-Member verwendet werden können.
- OPC Watch stellt nun eine Upload-Schaltfläche bereit, um eine Datei in einen Dateiknoten zu schreiben (der vorhandene Dateiinhalt wird vorher verworfen).
- Einführung der neuen Eigenschaft OpcNodeId.Resolution zur Konfiguration der Art und Weise, wie NodeIds aufgelöst werden, wenn keine benutzerdefinierten NodeIds verwendet werden.
- Einführung der neuen OpcNode-Ereignisse BeforeAdd und AfterAdd zur Behandlung von Bedingungen/Zuständen zu dem Zeitpunkt, zu dem ein Knoten zum Adressraum des Servers hinzugefügt wird oder wurde.
- Einführung der neuen Methode OpcNode.Implement, um die Erstellung des Knotens abzuschließen, bevor er zum ersten Mal verwendet werden kann.
- Unterstützung der Implementierung von Standard OPC UA Schnittstellen.
- Einführung von OpcOrderedObjectListNode als Standardimplementierung des OrderedListType.
- Einführung des abstrakten OpcOrderedObjectNode als Standardimplementierung des IOrderedObjectType / IOpcOrderedObjectNode.

BEHOBEN

- Problem mit nicht beachteten NodeIds von benutzerdefinierten Datentypkodierungen.
- Problem mit gecachten Anwendungszertifikaten, die einen benutzerdefinierten Anwendungsspeicherpfad verwenden.

2.25.0.0 (freigegeben am 22.03.2022)

NEU

- OPC UA Stack Types auf v1.04.368 aktualisiert.
- Die Server-definierten Datentypen-Knoten liefern nun keine Datentypdefinition mehr im Falle eines opaque Datentyps. Jede Leseanforderung führt zu BadAttributeIdInvalid und verhält sich nun genauso wie die eingebauten opaque Datentyp-Knoten.
- Einführung von IOpcData<T>, um eine Möglichkeit zu bieten, Datentypen abzuleiten, die programmatisch nicht vererbbar sind.
- Verschiedene Verbesserungen bei der Verarbeitung von Metadaten zu Datentypen, um die Leistung von Kodierungs-/Dekodierungsoperationen zu verbessern.
- Einführung neuer OpcClientSecurity-bezogener Eigenschaften zur Konfiguration des AuthenticationLevels und ImpersonationLevels, die der Client bei der Verbindung mit einem OPC Classic Server verwenden soll.

VERÄNDERT

- Default size of (auto) encoding mask from one bit to 32 bit to improve usability of the by default expected 32 bit encoding mask.

BEHOBEN

- Issue with (auto) bit number used when a data type members is marked as optional using

OpcDataTypeMemberSwitchAttribute.
- Issue with reloading the servers data type system after a connection has been lost and re-established.

2.24.0.0 (freigegeben am 09.03.2022)

NEU

- Einführung der neuen Schnittstelle IOpcAsyncMethodCommand zur Implementierung asynchroner Befehle.
- Einführung der neuen OpcMethodCommands.Execute-Methode zur Bereitstellung einer benutzerdefinierten generischen Befehlsmethode, die mit einem Befehlsobjekt ausgeführt werden kann.
- Einführung der neuen OpcMethodCommands.ExecuteAsync-Methode zur Bereitstellung einer benutzerdefinierten generischen Befehlsmethode, die asynchron mit einem Befehlsobjekt ausgeführt werden kann.
- Unterstützung von IOpcAsyncMethodCommand und Delegates, die auf eine Methode verweisen, die eine Task oder Task<T> zurückgibt. Beide werden asynchron beim Methodenaufruf ausgeführt.
- Einführung der neuen Eigenschaft OpcAutomatism.UseAsyncMethodCalls zur globalen Steuerung der Ausführung von Methodenaufrufen für jeden Server und alle seine NodeManager.
- Einführung der neuen Eigenschaft OpcNodeManager.UseAsyncMethodCalls zur lokalen Steuerung der Ausführung von Methodenaufrufen der vom NodeManager definierten Methoden.
- Unterstützung für OPC UA NumericRange Informationen mit der neuen OpcRange Struktur eingeführt.
- OpcReadNode wurde um die neue Eigenschaft Range erweitert, um das Lesen eines Bereichs von Elementen in einem Attribut zu unterstützen.
- Verbesserte OpcWriteNode mit neuer Range-Eigenschaft, um das Schreiben eines Bereichs von Elementen in einem Attribut zu unterstützen.
- Einführung zusätzlicher Überladungen der Methode OpcClient.ReadNode, um eine OpcRange zum Lesen eines Bereichs von Elementen in einem Attribut bereitzustellen.
- Einführung zusätzlicher Überladungen der Methode OpcClient.WriteNode, um einen OpcRange zum Schreiben eines Bereichs von Elementen in einem Attribut bereitzustellen.

VERÄNDERT

- Die Eigenschaft OpcNodeAccessToken.IndexRange ist jetzt veraltet, verwenden Sie stattdessen die neue Eigenschaft OpcNodeAccessToken.Range.
- Der Eigenschaftstyp von OpcReadVariableValueContext(<T>).Range wurde von Ua.NumericRange auf OpcRange geändert.
- Der Eigenschaftstyp von OpcWriteVariableValueContext(<T>).Range wurde von Ua.NumericRange in OpcRange geändert.
- Der Eigenschaftstyp von OpcReadPropertyValueContext(<T>).Range wurde von Ua.NumericRange auf OpcRange geändert.
- Der Eigenschaftstyp von OpcWritePropertyValueContext(<T>).Range wurde von Ua.NumericRange in OpcRange geändert.

BEHOBEN

- Probleme mit dem strikten XmlReflectionImporter, der in Mono/Xamarin beim Mappen von Klasseninformationen (einschließlich erwarteter Attribute und Eigenschaftstypen) verwendet wird.

2.23.0.0 (freigegeben am 02.03.2022)

NEU

- Einführung der neuen Eigenschaft Tag in OpcServiceCommand zur Zuweisung benutzerdefinierter Metadaten, die an einen bestimmten Befehl gebunden sind.
- Einführung der Load-, Parse- und TryParse-Methoden für OpcDataTypeDictionary, um verschiedene Möglichkeiten zu bieten, ein XML-serialisiertes Dictionary von Datentypen auf jede erdenkliche Weise zu konsumieren.
- Überarbeitung von OpcReadHistoryDetails, OpcReadNodeHistoryDetails, OpcReadNodeHistoryRawDetails und OpReadNodeHistoryModifiedDetails.
- Fehlende Dokumentation zu OpcReadHistoryDetails, OpcReadNodeHistoryDetails, OpcReadNodeHistoryRawDetails und OpReadNodeHistoryModifiedDetails hinzugefügt.
- Einführung der neuen Eigenschaft ReturnBounds in OpcReadNodeHistoryDetails.
- Einführung neuer ReadNode[s]History[Modified]-Methoden in OpcClient, um benutzerdefinierte OpcReadHistoryDetails zu unterstützen.

BEHOBEN

- OPC Classic-Verbindungen geben jetzt RPC_C_AUTHN_LEVEL_PKT_INTEGRITY für DCOM an, um die Härtungsänderungen für CVE-2021-26414 zu behandeln.

2.22.0.1 (freigegeben am 25.02.2022)

BEHOBEN

- Problem beim Aufruf von Acknowledge, Confirm und AddComment auf OpcCondition und OpcAcknowledgeableCondition Instanzen.

2.22.0.0 (freigegeben am 11.02.2022)

NEU

- Einführung der neuen Methode CreateArray in OpcArrayDimensions.
- Es wurden neue GetDataType-Methoden in OpcContext eingeführt, um kontextsensitive Datentypinformationen zu ermitteln.
- Unterstützung von OpcModelChangeEvent[Node] und OpcGeneralModelChangeEvent[Node] einschließlich der zugehörigen OpcModelChangeItem und OpcModelChangeActions eingeführt.
- Benutzerdefinierte Veröffentlichung/Behandlung von Modelländerungsereignissen unter Verwendung der neuen OpcModelChangeEventNode und OpcGeneralModelChangeEventNode.
- Einführung der Inline-Ereigniskomprimierung von Modelländerungsereignissen, wenn ein Ereignis-Snapshot eines OpcGeneralModelChangeEventNode bereitgestellt werden soll.

BEHOBEN

- Problem mit falscher Definition des ValueRanks bei Verwendung des OpcArguments-Konstruktors, der benutzerdefinierte Array-Dimensionen akzeptiert.
- Problem mit der neuen DataTypeDefinition, wenn ein benutzerdefinierter Aufzählungstyp definiert wird, bevor er zum ersten Mal verwendet wird.

2.21.0.0 (freigegeben am 01.02.2022)

NEU

- Client API - Vorbereitung zur OPC UA v1.04 kompatiblen Unterstützung von Datentypdefinitionen unter Verwendung des neuen Attributs DataTypeDefinition für Datentypknoten. Diese Erweiterung ist abwärtskompatibel zu OPC UA v1.03, da DataTypeDictionaries und DataTypeDescriptions einschließlich der DataTypeEncodings weiterhin zur Bestimmung der deklarierten und beschriebenen Datentypinformationen verwendet werden.
- Server API - Einführung der OPC UA v1.04 kompatiblen Unterstützung von Datentypdefinitionen unter Verwendung des neuen Attributs DataTypeDefinition für Datentypknoten. Diese Erweiterung ist abwärtskompatibel zu OPC UA v1.03, da DataTypeDictionaries und DataTypeDescriptions einschließlich der DataTypeEncodings weiterhin zur Deklaration und Beschreibung eines Datentyps verwendet werden.

2.20.4.0 (freigegeben am 14.01.2022)

NEU

- Clients werden nun mit dem Statuscode BadNodeIdUnknown benachrichtigt, wenn ein abonnierter Node später gelöscht wird.

2.20.3.0 (freigegeben am 04.01.2022)

BEHOBEN

- Problem mit OpcNamespace.Resolve(), sodass OpcNamespaceCollection.ResolveValue() nicht richtig funktionierte.
- IndexOutOfRangeException wenn ein nicht-generischer OpcAnalogItemNode im Server verwendet wird.
- Problem mit StackOverflowException wenn (implizit) ein Byte[] als Null-Verweis in einen ByteString konvertiert wird.

2.20.1.0 (freigegeben am 08.10.2021)

NEU

- Einführung des neuen OpcEnumAttribute, um die Definition von Enum-Membern zu steuern und um festzulegen ob für diese EnumStrings oder EnumValues verwendet werden. Zusätzlich bietet das OpcEnumAttribute Eigenschaften, um die bestehenden Automatismen je Enumeration selbst festzulegen.

BEHOBEN

- Problem beim Verbinden mit Server über 'https:' URLs unter .NET Framework.

2.20.2.0 (freigegeben am 20.10.2021)

BEHOBEN

- Problem mit doppelter Nennung von Fehlern in OpcException.Message.
- Problem mit OpcMethodNode bei Verwendung des falschen Callback-Ziels.

2.20.1.0 (freigegeben am 08.10.2021)

NEU

- Einführung des neuen OpcEnumAttribute, um die Definition von Enum-Membern zu steuern und um

festzulegen ob für diese EnumStrings oder EnumValues verwendet werden. Zusätzlich bietet das OpcEnumAttribute Eigenschaften, um die bestehenden Automatismen je Enumeration selbst festzulegen.

BEHOBEN

- Problem beim Verbinden mit Server über 'https:' URLs unter .NET Framework.

2.20.0.0 (freigegeben am 17.09.2021)

NEU

- OpcClient bietet jetzt Informationen zum Knotentyp für Ereignis-, Objekt- und Variablentypen.
- OpcNode bietet neue RegisterType und UnregisterType Methoden, um benutzerdefinierte Knotentypen zu verwalten.
- OpcNode bietet neue GetType- und GetTypeInfo-Methoden zur Ermittlung von Sytem.Type- und OpcNodeTypeInfo-Instanzen, die zur Definition/Beschreibung eines Knotentyps verwendet werden.
- OpcInstanceNode bietet neue GetNodeValue- und SetNodeValue-Methoden zum Ändern des Wertattributs eines bestimmten OpcVariableNode, der ein Kind des Instanzknotens ist.
- Die Ableitung der OpcObjectNode zur Definition einer benutzerdefinierten Knotentypdefinition führt auch zu inline erstellten Eigenschafts-/Variablenknoten für in der Unterklasse definierte Eigenschaften/Felder.
- Unterklassen von OpcTypeNode wurden eingeführt, um benutzerdefinierte Knotentypen zu definieren (OpcNodeType und OpcNodeType<T>).
- Einführung neuer Attribute zur Beschreibung benutzerdefinierter Knotentypen (OpcNodeTypeAttribute, OpcNodeTypeIgnoreMemberAttribute, OpcNodeTypeMembersAttribute, OpcNodeTypeOptionalMembersAttribute).
- Die Klasse OpcNodeTypeSystem wurde erweitert, um Informationen über aktuellen und standard Knotentypen des Systems bereitzustellen.
- Eigenschaften und Methoden in OpcNodeTypeInfo, um zusätzliche Informationen über den Typ und seine Kind-Nodes bereitzustellen.
- Die Datentypauflösung von OpcArgument wurde erweitert, um auch spät initialisierte benutzerdefinierte Datentypen zu unterstützen.

VERÄNDERT

- OpcInstanceNode bestimmt jetzt seine DefaultTypeDefinitionId anhand des Knotentyps, der zum Erstellen des Instanzknotens verwendet wird.
- OpcMemberOrigins definiert nun ein zusätzliches Enum-Mitglied „Method“ für Methoden.

BEHOBEN

- Problem mit spät definierten benutzerdefinierten Datentypen, die für Argumente in OpcMethodNodes verwendet werden.
- Problem mit BadNodeIdUnknown beim Zugriff auf die Input-/OutputArgumente von OpcMethodNodes, die erstellt wurden, bevor der zugehörige OpcNamespace initialisiert und im Server definiert wurde (z.B. bevor OpcNodeManager.CreateNodes ausgeführt wurde).

2.19.2.0 (freigegeben am 10.09.2021)

NEU

- Einführung der Unterstützung von IEnumerable<OpcName> in OpcNamePath.

- Es wurden neue OpcClient.GetType(s)-Methoden eingeführt, um alle mit einer NodeId verbundenen Typinformationen zu ermitteln.
- Es wurde die neue Methode OpcClient.IsSubtypeOf eingeführt, um festzustellen, ob ein Typ ein Subtyp eines anderen Typs ist.
- Es wurde die neue Methode IOpcNodeInfo.Child(OpcNamePath) eingeführt, um einen Kindknoten über einen Pfad von BrowseNames zu ermitteln.
- Einführung der neuen Eigenschaft ServiceName in OpcServiceRequest und OpcServiceResponse.
- Es wurden die neuen Methoden GetResults und GetResultExceptions in OpcServiceResponse eingeführt, um Anfrage- und befehlsbezogene Ausnahmeinformationen zu ermitteln.
- Einführung des neuen Ereignisses OpcServer.RequestJudgement zur Behandlung von Anforderungsproblemen, um zu entscheiden, ob eine Anfechtung (z.B. weil die Anforderung veraltet ist) abgelehnt werden soll.

VERÄNDERT

- Unnötige neue Zeilen in der String-Darstellung von OpcException-Instanzen wurden entfernt.

BEHOBEN

- Irreführende Verwendung desselben Basistyps (= der oberste Knotentyp) für den obersten Basistyp bei der Untersuchung der Typenhierarchie eines Knotentyps (z.B. BaseType von 'BaseEventType' war 'BaseEventType').
- Mögliche NullReferenceExceptions bei Verwendung der Methoden OpcClient.GetDataType und OpcClientGetNodeType mit einer Null-Referenz (Nothing in Visual Basic) anstelle einer gültigen Instanz der Klasse OpcNodeId.
- Problem, dass zu wenige HistoryValues zurückgegeben werden, wenn der Server keine Werte, sondern nur einen Fortsetzungspunkt in einer der Antworten gesendet hat.

2.19.1.0 (freigegeben am 01.09.2021)

NEU

- Die neue Methode GetValue in der OpcEvent Klasse ermöglicht nun in Unterklassen den Wert einer Eigenschaft einem Node-Wert des Events zuzuordnen und das ohne Kenntnis des entsprechenden NamespaceIndexes für einen vollqualifizierten BrowseName. Hierzu wird als Index der Namespace der jeweiligen Typ-Definition verwendet, für die diese Eigenschaft implementiert wird. Ebenso kann die Methode wie die bekannte DataStore.Get-Methode verwendet werden und somit auch weiterhin explizit der vollqualifizierte BrowseName übergeben werden. Dadurch muss nicht länger der NamespaceIndex beim Zugriff auf Event-Daten bekannt sein.

VERÄNDERT

- **MÖGLICHER BRUCH:** IOpcReadOnlyNodeDataStore erfordert nun auch die Implementierung von IOpcNamespaceResolver.

BEHOBEN

- Problem mit der Auflösung des entsprechenden benutzerdefinierten OpcEvent-Typs/der entsprechenden benutzerdefinierten Instanz für ein benutzerdefiniertes Ereignis, das mit der vollqualifizierten NodeId des Ereignistyps deklariert wurde.
- Problem mit Hängern bei der Verbindung zu OPC Classic Servern.

2.19.0.0 (freigegeben am 27.08.2021)

BREAKING CHANGE: Clients und Server seit v2.17.0.0 greifen nicht gemäß der OPC UA Spezifikation auf Dateiknoten zu bzw. stellen diese bereit. Sie verwenden ein Byte-Array anstelle von ByteString zum Lesen/Schreiben von Dateiinhalten. **Dieser Fehler wurde in dieser Version korrigiert.**

NEU

- Die Auflösung von Knotentypen und Datentypen wurde verbessert, um auch erweiterte (= aufgelöste) Knotenbezeichner zu unterstützen.

VERÄNDERT

- **BRUCH:** OpcFileReadMethodNode.FileReadCallback und OpcFileReadMethodNode.FileReadExCallback geben jetzt einen ByteString statt eines Byte-Arrays zurück.
- **BRUCH:** OpcFileWriteMethodNode.FileWriteCallback und OpcFileWriteMethodNode.FileWriteExCallback akzeptieren jetzt einen ByteString für den Datenparameter anstelle eines Byte-Arrays.
- Parameter von OpcFilterOperand.OfType von „value : object“ in „typeId : OpcNodeId“ geändert, um sicherzustellen, dass ein „OfType“-Operand wie erwartet ausgewertet wird (= unter Verwendung einer OpcNodeId).

BEHOBEN

- **BRUCH:** Regressionsproblem bei der Verwendung von OpcFileNode-Zugriff mit Lese- und Schreiboperationen nach der Erzwingung der Verwendung von ByteString für ByteString und Byte[] für Byte[] durch Foundation Stack anstelle der Verwendung von ByteString für Byte[].
- Problem, dass keine historischen Werte zurückgegeben werden, wenn der Server den Statuscode „GoodNoData“ mit einem Fortsetzungspunkt meldet.
- Problem mit unendlichen historischen Werten, die zurückgegeben werden, wenn der Server die Wiederverwendung von Fortsetzungspunkten erlaubt.

2.18.5.0 (freigegeben am 13.08.2021)

NEU

- Es wurde die Unterstützung der kodierungsbasierten Abfrage von Datentypen durch OpcClient.GetDataType mit z.B. OpcValue.DataTypeId eingeführt.
- Verbesserte Verarbeitung von (strukturierten) Datentypinformationen, um doppelte Datentypdeklarationen zu unterstützen und auf Datentypbeschreibungen umzuleiten, wenn darauf verwiesen wird.
- Vereinfachte Definition von Eigenschaftsknoten ohne manuelles Hinzufügen des Knotens zu den Kindern eines übergeordneten Knotens.

BEHOBEN

- Regression in v2.18.4.0 mit OpcException+BadServerNotConnected bei aktivierter Option OpcClient.UseDynamic während der Verbindung mit einem Server.
- Probleme mit Siemens OPC UA Servern, die eine benutzerdefinierte Serverschnittstelle mit benutzerdefinierten strukturierten Datentypen verwenden. Diese Lösung erfordert die „Aktivierung der Standard-SIMATIC-Serverschnittstelle“, um die benutzerdefinierte Serverschnittstelleneinrichtung und die verwendeten Datentypen zu unterstützen.

2.18.4.0 (freigegeben am 09.08.2021)

NEU

- Unterstützung von Expression<TDelegate> kompilierten Delegaten in OpcMethodNodes eingeführt.
- Einführung eines Indexers, der ein Int32 verwendet, um ein spezielles OpcDataField in OpcDataObjects zu adressieren.
- Das Ereignis OpcClient.SubscriptionsChanged wird jetzt ausgelöst, wenn OpcClient.Subscriptions gelöscht wird.
- Veralteter Code, der in .NET 5.0 nicht mehr unterstützt wird, wurde entfernt.
- Verbesserte Ausnahmebehandlung/Meldung beim Importieren eines UANodeSet.

BEHOBEN

- Problem mit spät geladenen benutzerdefinierten Typinformationen, nachdem das Ereignis OpcClient.Connected ausgelöst wurde.
- Problem mit der erweiterten byte[]-Unterstützung, das dazu führte, dass die sbyte[]-Unterstützung als variabler Knotenwert nicht mehr funktionierte (Grund für das Problem: .NET unterscheidet nicht zwischen vorzeichenbehafteten und vorzeichenlosen Typen bei der Verwendung von Mustervergleichen).
- Problem beim Schreiben von Byte[]-Werten in Variablenknoten, da der Stack der Foundation die Verwendung von Variant erzwingt, um Byte-Arrays als Byte[] anstelle eines ByteString zu verarbeiten. Dies führt jedoch zu falschen Typvalidierungen beim Schreiben eines Variablenknotens mit einem Byte-Array als Datentyp/Wert. Gelöst wurde das durch die Berücksichtigung des nachfolgenden Stack-Codes.

2.18.3.0 (freigegeben am 28.06.2021)

NEU

- Erweiterte Exception-Details über die Inner-Exceptions wie auch Aggregate-Exceptions, um weitere Informationen bei OPC Classic Verbindungsproblemen bereitzustellen, wenn der OPC Classic Wrapper Server im gleichen Prozess ausgeführt wird.
- Einführung weiterer OpcNodeIdComparison-Optionen: IgnoreNamespaceIndex und IgnoreNamespaceIndexIgnoreCase.

VERÄNDERT

- OpcClassicDiscoveryClient verwendet nun die OpcException um Stack-bezogene Exceptions bereitzustellen.

BEHOBEN

- Probleme mit der OpcServerInfo, wenn das SDK in einer „Bundled Assemblies“-Umgebung verwendet wird.
- Probleme beim Verbinden zu OPC Classic Server die auf einem anderen Rechner ausgeführt werden.
- Probleme mit fehldenen Nodes beim Browsing von OPC Classic Servern mit flacher Organisation der Nodes.
- Probleme bei der Übertragung von Byte-Arrays an Methoden-Nodes, aufgrund der vom Stack der Foundation verwendeten Logik standardmässig Byte-Arrays als ByteStrings zu übertragen.

2.18.2.0 (freigegeben am 17.06.2021)

NEU

- Es wurden neue Translate-Methoden in OpcServerGlobalization eingeführt, um Mechanismen zur Lokalisierung von benutzerdefinierten Ressourcen bereitzustellen.
- Unterstützung von Enum-Arrays eingeführt, die im OPC Stack v1.03 fehlte.
- OpcNodeId.Equals Methoden die für den benutzerdefinierten Vergleich von NodeIds anhand vordefinierter Vergleichsmethoden über OpcNodeIdComparison verwendet werden können.

VERÄNDERT

- OpcNodeId vergleicht nun NodeIds entsprechend der Spezifikation von OPC UA unter Berücksichtigung der Groß- und Kleinschreibung den Identifier und den Namespace.

BEHOBEN

- Problem mit fehlenden bevorzugten Gebietsschemata in OpcSession nach dem Aktivieren einer Sitzung. Die Locales wurden nur intern aktualisiert und verwendet.
- Probleme mit ServerHalted-Ausnahmen beim Zugriff auf Eigenschaften einer Instanz der Klasse OpcServer vor dem Starten der Instanz.

2.18.1.0 (freigegeben am 02.06.2021)

NEU

- Der fehlende Setter der Eigenschaft OpcBrowseNode.Degree wurde hinzugefügt.
- Die neue Eigenschaft OpcClient.Interfaces wurde zur Konfiguration der Kommunikation über die OPC Classic Schnittstellen eingeführt.
- Die neue Klasse OpcClassicDataAccessOptions stellt Optionen bereit, um die Art der Organization des Servers und ob der Server den BrowseTo-Dienst unterstützt festzulegen.

BEHOBEN

- Problem unter Mono beim Verbinden zu einen OPC UA Server, welcher kein Zertifikat bereitstellt.
- Problem fehlender Subscriptiob-basierter Notifications nach einem automatischen Wiederaufbau der Verbindung.
- Problem bei der Verwendung eines byte-Arrays in einem OpcValue, welches als ByteString anstelle als Byte-Array vom zugrundeliegenden Stack kodiert wurde.
- Problem bei der Kodierung des Value-Attributs bei nicht-generischen OpcVariableNodes, wenn ein Array von ByteString-Werten verwendet wird.

2.18.0.0 (freigegeben am 26.05.2021)

NEU

- Verbesserungen für Szenarien in denen Byte[], ByteString und ByteString[] oder einzelne von diesen als Datentyp von Methoden-Argument or Variablen-Nodes verwendet werden.
- Unterstützung der OpcDataTypeInfo.Documentation-Eigenschaft anhand des OpcDataTypeNode.Description Attributes.
- Die neuen Methoden OpcReferenceTypes.GetReferenceTypeName(OpcNodeId) und GetReferenceType(OpcName) wurden eingeführt.
- Vollständige Überarbeitung und Erweiterung der Typen OpcRelativePath und OpcRelativePathElement.
- Einführung der Unterstützung der Browsepfad-Übersetzung unter Verwendung des TranslateBrowsePath Dienstes.
- OpcClient unterstützt nun die Browsepfad-Übersetzung über die neuen TranslatePath und

TranslatePaths Methoden.

VERÄNDERT

- Format von ByteString wurde von Base64-basierter Kodierung in Hexadezimaler Formatierung (= „X2“) geändert, um der binär kodierten Darstellung eines ByteStrings der Foundation zu entsprechen.

BEHOBEN

- Falsch initialisierte ArrayDimensions- und ValueRank-Attribute von Methodenargumenten und Variablenknoten, die Byte[], ByteString oder ByteString[] oder einen davon als Datentyp verwenden.
- Falsche Initialwerte von DataTypeDictionary-Knoten bezüglich verwendeter ArrayDimensions- und ValueRank-Attribut nach Einführung des neuen expliziten ByteString-Datentyps.
- Eine vermeidbare NullReferenceException beim Initialisieren benutzerdefinierter Datentypen wird nicht mehr ausgelöst.
- Probleme mit falsch bestimmten OpcArgument-Metadaten, die zum Definieren von Array-basierten Methodenargumenten verwendet werden.
- Probleme bei der Untersuchung von System.Type Informationen bei „out“-Parametern, welche als „by-reference“ im Delegaten eines Method-Nodes verwendet werden.
- Probleme beim Importieren von UANodeSets-Referenzen, wenn diese über Aliase definiert werden.
- Problem mit fehlenden Type-Informationen bei der Untersuchung von Felder benutzerdefinierter Datentypen, die vom Typen eines Arrays sind. Hier fehlte dann die TypeId des ElementTypen im ArrayType des Feldes.
- Probleme in Anwendungen mit Third-Party-Komponenten, welche Attribute verwenden, die nicht ladbar sind. Dies löst bekannte TypeLoadExceptions die bei der Untersuchung von benutzerdefinierten Datentypen / Eventtypen auftreten konnten.

2.17.0.0 (freigegeben am 04.05.2021)

NEU

- Unterstützung für benutzerdefinierte Implementierungen der OpcFileInfo Klasse.
- Erweiterung des OpcMethodContexts mit Verweis zum Owner, um Zugriff auf die zugrundeliegende OpcServer-Instanz zu erhalten.
- Erweiterung der (En)Kodierung zur Unterstützung von strukturierten Datentypen anhand ihrer Basistyp-Informationen.
- Unterstützung von Null-Referenzen bei Verwendung von OpcNamespace.Create.
- Einführung des expliziten Datentypen ByteString, um die Differenzierung zwischen Byte-Array und ByteString-Werten zu erleichtern.

BEHOBEN

- Problem mit Servern die Null-Referenzen als Teil ihres NamespaceArrays verwenden.
- Problem mit zurückgesetzter OpcNode.Description im Falle von z.B. OpcDataVariableNode<T> unter Verwendung eines benutzerdefinierten Datentypen.
- Problem mit nicht initialisierten Event-Nodes bei Verwendung außerhalb eines AddressSpace-Szenarios („off-address-space“) z.B. bei „leichtgewichtigen“ Events.
- Problem mit automatischer Firewall-Awareness und den dabei verwendeten DNS-sicheren Hostnamen. Um eine neue Sitzung in diesem Fall zu erstellen wird nun die via Discovery ermittelte „raw“ Endpunkt-URL verwendet.
- Problem mit TypeInitializationException bei der Prüfung auf Verwendung der Unity-Runtime in manchen dynamischen Assembly-Lade-Szenarien.

- Problem mit InvalidOperationException in Foundations BufferManager immer dann, wenn ein gesperrte Puffer freigegeben werden soll, jedoch nicht zuvor entsperrt wurde.
- Problem mit nicht aufgelösten Datentyp-Informationen in Argumenten von Methoden-Nodes, wenn erweiterte NodeIds für den Datentypen verwendet werden.
- Problem mit nicht aufgelöster Datentype-Kodierung im Typsystem des Server, wenn erweiterte NodeIds für die Datentypen-Kodierung verwendet werden.
- Problem beim Vergleich von erweiterten NodeIds mit dem selben Identifier jedoch unterschiedlichen Namespaces.
- Problem beim Ermitteln von Typ-Informationen von benutzerdefinierten Datentypen mit den selben Namen jedoch in unterschiedlichen Namespaces.
- Problem mit NullReferenceExceptions beim Versuch eine Resource zu übersetzen, ohne zuvor Resource-Informationen hinterlegt zu haben.
- Problem mit fehlenden benutzerdefinierten Datentypen, wenn diese über mehrere (dynamische) NodeManager definiert werden.
- Problem mit der Vererbung des Standard-ReferenceTypen bei Verwendung von OpcFolderNodes. Wodurch der Standard-ReferenceType Organizes, nach einem Stack-Update, nicht länger für dessen direkten Kind-Nodes übernommen wurde.
- Weiterführendes Problem bei der Deklaration/Verwendung von Variablen-Nodes mit ExpandedNodeId/NodeIds als Datentypen.
- Threading-Problem beim asynchronen Start mehrerer OpcServer-Instanzen.
- Falscher Wert de ArrayDimensions-Attributs der Input-/OutputArgument Nodes von OpcMethodNodes.

2.16.1.0 (freigegeben am 21.04.2021)

NEU

- OpcNominalNodeIdFactory wurde erweitert, um zusätzliche benutzerdefinierte Logik auf den verwendeten OpcName anzuwenden, anhand dessen die OpcNodeId erstellt wird.

BEHOBEN

- Problem mit weiterhin existierenden OpcSubscription-Instanzen unter den OpcClient.Subscriptions, obwohl der Client nicht länger verbunden ist.

2.16.0.0 (freigegeben am 20.04.2021)

NEU

- Unterstützung für benutzerdefinierte OpcFileNode-Quellen über die neue IOpcFileInfo Schnittstelle.

BEHOBEN

- Problem beim Deklarieren/Zugreifen auf Variable-Nodes welche ExpandedNodeId als Datentyp verwenden.
- Problem mit FileNotFoundExceptions, wenn Drittanbieter Assemblies verwendet werden, während die dynamische Typregistrierung aktiviert ist.

2.15.0.0 (freigegeben am 31.03.2021)

NEU

- Opc.UaFx.Client.OpcSubscription.Client-Eigenschaft wurde eingeführt, um Zugriff auf den Client zu

erhalten, welcher der Besitzer einer Subscription ist.
- OpcServer.Binding.Addresses-Eigenschaft wurde eingeführt, um explizite IP Adressen der Netwerkschnittstellen zu binden und auf diesen zu lauschen.
- OpcNodeCollection.TryGet-Methode, um zu versuchen einen Node anhand seines Node-Identifiers zu ermitteln.
- OpcNodeManager.GetNode-Methoden, um Nodes anhand ihres Node-Identifiers oder über einen Pfad, welcher eine Sequenz aus aneinandergefügten Browse- und Symbolic-Names der Nodes auf dem „Pfad“ zum Node jeweils getrennt durch einen Slash ('/'), beschreibt.

BEHOBEN

- Problem beim Erstellen von Property-Nodes ohne Parent-Node.

2.14.0.0 (freigegeben am 04.03.2021)

NEU

- Einführung der statischen Methode OpcName.IsNullOrEmpty, welche bei der Auswertung die Namespace Informationen des OpcNames ignoriert.
- Einführung der statischen Methode OpcNodeId.IsNullOrEmpty, welche bei der Auswertung die Namespace Informationen der OpcNodeIds ignoriert.
- Einführung der OpcNamespace.GetId(object) Method, um Node Identifier zu ermitteln, ohne den Typen des Wertes zu kennen.
- Einführung der dynamischen Auflösung von Datentyp-Informationen ohne diese zuvor dem Datentypen-System mitzuteilen (= zu registrieren).
- Einführung von zusätzlichen nicht-generischen OpcData.RegisterType Methoden, um nicht länger die Verwendung der Methoden mit generischen Typ-Parameter zu erzwingen.
- Einführung der neuen OpcDataTypeNode.Create(…) Methoden, um einen nicht-generischen Weg bereitzustellen, um OpcDataTypeNodes Instanzen zu erstellen.

VERÄNDERT

- Doppelte Verweise auf DataTypeNodes, bei Verwendung der OpcAddressSpace Klasse, wurden entfernt.

BEHOBEN

- Problem mit fehlerhaften oder „zu vielen“ Subscriptions, welches den OpcClient daran hinderte sich automatisch, nach Verlust einer Verbindung, wieder neu zu verbinden.
- Problem mit inline initialisierten OpcDataTypeNodes bei Verwendung der dynamischen Auflösung von Datentyp-Informationen.
- Problem mit fehlenden Typ-Referenzen bei benutzerdefinierten Enumerationen im globalen Datentyp-System.

2.13.0.0 (freigegeben am 01.03.2021)

NEU

- Verbesserte Auflösung des Data Type Attributs mit Unterstützung von „expanded node identifiers“.
- Verbesserte dynamische Data Type De-/Kodierung zur Unterstützung von Unterklassen und kaskadierten benutzerdefinierten strukturierten Datentypen.
- Das OpcDataTypeIgnoreMemberAttribute wurde eingeführt, um explizit Mitglieder eines Typen von der De-/Kodierung auszuschließen.

- Die Eigenschaften OpcDataTypeAttribute.NamespaceUri und OpcDataTypeAttribute.UseDataTypeDescription wurden eingeführt, um den Zielnamespace der zu verwendenden Data Type Dictionary eines Datentypen zu definieren und um festzulegen, ob der Datentyp eine Beschreibung in einem benutzerdefinierten Data Type Dictionary benötigt.
- Unterstützung von benutzerdefinierten Zielnamespaces für benutzerdefinierte Enumerationen.
- Unterstützung von eingebetteten benutzerdefinierten Datentypen beim dynamischen Datenaustausch über Container-Typen wie Variant und/oder ExtensionObject.

BEHOBEN

- Problem mit fehlenden Zielnamespace-Verweisen im inline definierten Data Type Dictionary, wenn der benutzerdefinierte Datentyp ohne Zielnamespace-Informationen definiert wurde.

2.12.3.0 (freigegeben am 17.02.2021)

NEU

- Verbesserte Unterstützung von IL2CPP bezüglich Problemen bei der Auswertung von eingebetteten String-Werten.

BEHOBEN

- Probleme bei der Verwendung der standard Zertifikat-Probing-Pfade unter Unity mit UWP.
- Problem beim Aufruf von Method-Nodes nach der Einführung der Lokalisierung von Argumenten.

2.12.2.0 (freigegeben am 15.02.2021)

NEU

- Unterstützung zur Übersetzung von Argumenten von Methoden-Nodes.

BEHOBEN

- Probleme beim Zugriff auf lokalisierte Informationen mit dem Client mit einer neutralen Kultur.
- Problem Null-Referenzen (Nothing in Visual Basic), wenn lokalisierbare Attribute ohne Übersetzung gelesen werden.

2.12.1.0 (freigegeben am 11.02.2021)

NEU

- Einführung der neuen Klasse OpcNodeExportOptions, zur Konfiguration der Informationen die beim Export von Nodes in ein Node Set exportiert werden sollen.
- OpcNodeManager.ExportNodes(OpcNodeExportOptions) Methoden Überladung zur Definition benutzerdefinierter Exportoptionen.
- Einführung der neuen Klasse OpcNodeGlobalization, um die Ressourcen zu verwalten, die zur Lokalisierung von Attributen einer bestimmten Node verwendet werden sollen.
- Die Eigenschaften OpcNode.DisplayNames und OpcNode.Descriptions, um Instanzen bereitzustellen, über die die Lokalisierung der lokalisierbaren Attribute einer Node auf Node-Level gesteuert werden kann.

VERÄNDERT

- Bidirektionale Verweise zwischen DataTypeDescription-, DataTypeEncoding- und DataType-Nodes werden jetzt unidirektional exportiert (unter Verwendung der Standard Exportoptionen).

BEHOBEN

- Problem mit automatisch ermittelten Server-Informationen anhand von Prozess-bezogenen Versions-Informationen, welche die Auswertung der ProductVersion und FileVersion mit einschloss.
- Problem mit dem Überschreiben des OpcNode.SymbolicName, wenn eine Node importiert wird, welche einen anderen Wert für die SymbolicName Eigenschaft als für das DispalyName Attribut verwendet.
- Problem mit nicht länger mehrzeilig Base64 kodierten DataTypeDictionary (unter Verwendung der Standard Exportoptionen).
- Problem mit entfernten ParentNodeId Attributen in NodeSets (unter Verwendung der Standard Exportoptionen).
- Problem mit umsortierten Nodes (anhand ihrer NodeId), wenn diese in ein NodeSet exportiert werden. Ab sofort werden diese in der Reihenfolge ihrer Definition / Implementierung exportiert.
- Problem mit fehlenden SuperTypen beim Import von Nodes, welche zuvor in einen Server importiert und wieder exportiert wurden (aufgrund der Neuordnung der Nodes).

2.12.0.0 (freigegeben am 04.02.2021)

NEU

- Neue Namespaces Eigenschaft in IOpcApplicationInstance und OpcApplicationInstance, um die verwendeten Namespace bereitzustellen.
- Neue Namespaces Eigenschaft im OpcClient, um die Namespaces des verbundenen Servers abzubilden.
- Neue Namespaces Eigenschaft im OpcServer, um die Namespaces bereitzustellen, die von den Node Managern des Servers verwendet werden um den Adressraum zu organisieren.
- Neue Namespaces Eigenschaft in OpcContext, um die im aktuellen Kontext gültigen Namespaces bereitzustellen.
- Die Schnittstelle IOpcNamespaceResolver wurde in der OpcNamespaceCollection und OpcReadOnlyNamespaceCollection implementiert.
- Resolve Methode in IOpcNamespaceResolver, um einen anderen (= „online“) Namespace anhand der Metadata eines (offline / lokal) definierten Namespaces aufzulösen.
- OpcNamespace stellt jetzt immer dann eine Uri bereit, wenn der OpcNamespace.Value eine gültige Uri darstellt.
- Verbindungen zu OPC Classic Servern durch einen inline verwendeten Wrapper-Server werden jetzt dadurch gesichert, indem die gleiche Benutzeridentität verwendet wird, die unter OpcClient.Security.UserIdentity eingestellt wurde.
- OpcClient und OpcClassicDiscoveryClient unterstützen jetzt die Verwendung einer Hostidentität zur DCOM-Authentifizierung. Hinweis: Im Falle des OpcClients wird die unter OpcClient.Security.UserIdentity eingestellte Identität ebenfalls zur DCOM-Authentifizierung verwendet.
- Neue Scope Eigenschaft im OpcNamespace, um den Kontext des Namespaces zu beschreiben, in dem dieser erstellt wurde und innerhalb dessen auch die Metadaten des Namespaces gelten.
- Verbesserte Geschwindigkeit bei der Initialisierung von OpcNodes während des Starts des OpcServers.
- Die Verwendung von OpcNodeManager.ctor(OpcNamespace, ...) mit einen Namespace, der einen expliziten Namespace Index besitzt, führt zu einer ArgumentException, da es die Aufgabe des Servers ist den verwendeten Namespaces den zugehörigen Namespace-Index zuzuweisen.
- Die OpcNamespace Instanzen im OpcNodeManager werden jetzt mit den OpcNamespace Instanzen der neuen OpcServer.Namespaces Eigenschaft synchronisiert. Zu beachten ist, dass in der Hochfahrphase des Servers sich deshalb die Instanzen in OpcNodeManager.Namespaces ändern

können. Spätestens jedoch beim Aufruf von z.B. OpcNodeManager.CreateNodes(...) sind alle OpcNodeManager.Namespaces bereits mit den OpcServer.Namespaces synchronisiert.

VERÄNDERT

- Der OpcNamespace.Default Namespace stellt jetzt den zum Standard-Namespace gehörigen Value sowie die Uri bereit, unter der Namespace der Foundation identifiziert werden kann.
- Abgekündigte Methoden in der IOpcNamespaceResolver sowie deren Implementierung in OpcClient, OpcServer, OpcContext und OpcNodeManager wurden entfernt.
- OpcClient.State Übergänge finden nicht länger statt (unter Verwendung des standardmäßig aktivierten Reconnect-Mechanismus), wenn während eines Wiederverbindungsversuchs z.B. ReadNode aufgerufen wird.
- Im Falle einer unterbrochenen Verbindung versucht der OpcClient (standardmäßig) die Verbindung wiederherzustellen und dabei die zuvor verwendete Sitzung wiederzuverwenden. Ist die Wiederverwendung der vorherigen Sitzung (nach Wiederaufbau der Verbindung) möglich, wird der OpcClient (wie bisher) das Reconnected-Ereignis auslösen. Im Falle, dass der Client gezwungen ist, eine neue Sitzung zu erstellen (z.B. weil diese abgelaufen ist), dann wird der OpcClient jetzt das Connected-Ereignis anstelle des Reconnected-Ereignisses auslösen. Hierdurch soll eine Möglichkeit geboten werden im Benutzercode diese Situationen zu unterscheiden.
- Vergleiche von OpcNodeId und OpcNamespace gegenüber Null-Verweisen verhalten sich jetzt wie jede andere typische .NET API (dieses Verhalten unterscheidet sich dem des Stacks der Foundation).

BEHOBEN

- Problem mit „collapsed“ Endpunkt-URLs beim Discover von Servern, welche z.B. "opc.tcp://:4840" als Endpunkt-URL liefern.
- Problem mit nicht aufgelösten Node-Identifiern, wenn manuell erstellte Monitored-Items einer Subscription zugewiesen werden.
- Problem mit „über-aufgelösten“ Node-Identifiern durch die Auflösung zu vollständig neuen Node-Identifiern (wie bisher) nur ohne die Namespace Informationen des ursprünglichen Node-Identifiers immer wieder zu überschreiben.
- Problem im OPC Classic Stack mit zunehmender Anzahl an Threads (insbesondere in MTA-Anwendungen), sobald einmal die Verbindung zum OPC Classic Server verloren ging.
- Problem bei der Definition von benutzerdefinierten Datentypen ohne eine explizite NamespaceUri in dessen DataTypeEncoding festzulegen.
- Problem mit dem Verlust von OpcNamespace-Informationen nachdem ein einmal initialisierter Node-Identifier anhand eines Foundation Node-Identifiers erneut initialisiert wird, da diese sich zwischenzeitlich geändert hat.
- Problem in Anwendungen, in denen mehrere Server oder Client und Server gleichermaßen im gleichen Prozess ausgeführt werden, sodass alle dabei beteiligten Namespaces nicht in jeden Fall der entsprechendend Anwendung entsprachen. Dieses Problem wurde nun gelöst, in dem nicht länger indirekt der Namepsace eines existierenden Node-Identifiers noch einer OpcNamespace-Instanz aufgelöst werden. Sollte das dennoch bei einen Benutzer gewünscht sein, dann muss die entsprechende Logik selbst implementiert werden.

2.11.5.0 (freigegeben am 21.12.2020)

NEU

- Optimierung der verwendeten Aliases in exportierten UA NodeSets (nicht verwendete werden beim Erstellen des UA NodeSets entfernt).
- Redundante Informationen in DisplayName- und Description-Attribut werden jetzt entfernt. In Falle,

dass beide Attribute den gleichen Wert besitzen, wird das Description Attribut nicht länger bei der Erstellung des UA NodeSets (bei gleichen Werten) exportiert.
- Redundante Informationen im ParentNodeId-Attribut werden nicht länger bei der Erstellung des UA NodeSets exportiert, im Falle, dass der ParentNode bereits durch eine Referenz über die ReferenceTypeId des Kind-Nodes referenziert wird.

BEHOBEN

- Problem mit NullReferenceExceptions im OpcServer, wenn versucht wird einen neuen Node (durch einen Client) anzulegen, während auf einen ParentNode mit einer unbekannten / ungültigen NodeId verwiesen wird.
- Problem mit fehlenden Node-Verweisen (zu Nodes die in einen anderen Node Manager definiert werden) in UA NodeSets beim Exportieren von Nodes deren NodeId implizit festgelegt wurde.
- Problem beim Export von Nodes welche aufeinander über das ParentNodeId-Attribut verweisen und zugleich ein expliziter Node-Verweis (auf die ParentNodeId wird nun in solchen Fällen verzichtet; wenn child.ReferenceTypeId [= die Standardreferenz zwischen Eltern und Kind-Nodes] gleich der zu beschreibenden Referenz im UA NodeSet ist) besteht. Der Import des bisher resultierten UA NodeSets führte zu einen Adressraum mit mehrfachen Verweisen zwischen den Nodes (welche durch das ParentNodeId Attribut und den Referenzen beschrieben werden).

2.11.4.0 (freigegeben am 15.12.2020)

NEU

- Es wurde die OpcContext.Owner Eigenschaft eingeführt, welche auf IOpcApplicationInstance Instanzen wie OpcClient oder OpcServer verweist, zu den der Kontext gehört (soweit verfügbar).
- Es wurde die OpcException.Context Eigenschaft eingeführt, welche eine Instanz der OpcContext Klasse bereitstellt, über die weitere kontextabhängige Informationen bezüglich der geworfenen/behandelten Ausnahme abgerufen werden können.

BEHOBEN

- Problem bei der Verwendung von abstrakten Methoden für Methoden-Nodes (Ursache ist ein Fehler im .NET Framework unter Verwendung der GetCustomAttribute<T> Erweiterungsmethoden).

2.11.3.1 (freigegeben am 27.11.2020)

BEHOBEN

- Problem beim Verbinden mit Servern, die Endpunkt URLs mit Groß-/Kleinschreibung unterscheiden und URLs mit der Endung '/' nicht unterstützen.

2.11.3.0 (freigegeben am 23.11.2020)

NEU

- Fehlende OpcEventDataSetItem.ClientID Eigenschaft wurde ergänzt.
- Fehlende Code Dokumentation der OpcSecurityPolicy, OpcSecurityMode und OpcSecurityAlgorithm wurden ergänzt.
- Einführung der PreferredLocales Eigenschaft in OpcContext, OpcOperationContext und OpcSession.
- Einführung der neuen OpcNode.Read/WriteDisplayNameCallback Eigenschaften.
- Einführung der OpcServer.GetSession(sessionId) Methode zum Abfragen einer bestimmten Sitzung über deren Session ID.

- Einführung der OpcServer.Globalization Eigenschaft zur besseren Bereitstellung von lokalisierten Informationen und Daten.

VERÄNDERT

- Verwendeter Datentypen der OpcNode.Read/WriteDescriptionCallback Eigenschaft wurde von String auf OpcText geändert.
- OpcDataChangeDataSetItem.ClientID von Int32 zu Int64 geändert, um dem bei OpcMonitoredItem.ClientID verwendeten Typen zu entsprechen.

BEHOBEN

- Problem mit fehlender OpcNotification.Data, wenn OpcNotification.GetDataChanges() verwendet wird, ohne zuvor auf die OpcNotification.Data zugegriffen zu haben.
- Problem bei der Typ-Initialisierung in OpcClient unter Verwendung von Unity mit IL2CPP wurden Typen in einer abweichenden Reihenfolge initialisiert.

2.11.2.0 (freigegeben am 10.11.2020)

NEU

- Experimentelle OpcNodeManager.ExportNodes Methode zur Generierung von OpcNodeSet Instanzen anhand the Adressraums der von Node Managern definiert wird. Nutzen Sie die Methode je nach Anwendungsfall und geben Sie uns Feedback im Falle von Problemen oder gewünschten Erweiterungen.

BEHOBEN

- Problem welches zum Hängen bleiben von OpcClient.Dispose() führen kann während noch eine Subscription aktiv ist.
- Möglichkeit von Ausnahmen beim Aufruf von OpcClient.Dispose und OpcServer.Dispose.
- Problem bei der Validierung von Lizenzinformationen unter Verwendung von IL2CPP und .NET Standard 2.0 als Scripting Backed in Unity.
- Problem welches die korrekte Verwendung eines existierenden SynchronizationContexts zur Verarbeitung von Notifications im selben Kontext, in dem eine Subscription erstellt wurde, verhinderte.
- Problem bei Hinzufügen von weitern Node-Referenzen (Verweise wurde nicht korrekt in den Adressraum übernommen).

2.11.1.0 (freigegeben am 05.11.2020)

NEU

- Der OpcServer kann jetzt auch gestoppt werden, wenn dieser pausiert wurde.
- Eigenschaft zur Konfiguration der TimestampsToReturn für alle ReadNodesHistory-Anfragen über OpcClient.Services.ReadNodesHistory.TimestampsToReturn.
- Eigenschaft zum Abrufen der ApplicationUri, welche als „Subject Alternate Name“ Erweiterung in Zeritikaten enthalten ist: OpcCertificateInfo.ApplicationUri
- Eigenschaft OpcClientSecurity.AutoUpgradeEndpointPolicy zum Aktivieren der Endpunkt-Auswahl basierend auf einer Überstimmung mit der Konfiguration (= Standard) und einer „nächst-besten Technik“ zur Bestimmung eines Endpunktes, der mindestens die konfigurierte Policy gewährleistet.

VERÄNDERT

- Der Konstruktor der OpcCertificateJudgementEventArgs prüft nicht länger, ob eines der übergebenen Zertifikate eine Null-Referenz (Nothing in Visual Basic) ist, weil manche Server keinerlei Zertifikatdaten für Endpunkte mit der Policy „None“ (= Mode und Algorithm) verwenden.

BEHOBEN

- Problem mit nicht länger verfügbaren Notifications in Subscriptions / Monitored Items, wenn diese nicht (länger) in der aktuellen Sitzung erstellt / aktiv sind.
- Problem mit fehlenden / fehlerhaften Security Policies nach dem Neustart von OpcServer Instanzen.

2.11.0.0 (freigegeben am 06.10.2020)

NEU

- Neue Methode OpcServer.Suspend(), um den Server in den Status Suspended zu versetzen.
- Neue Methode OpcServer.Resume(), um den Server aus dem Status Suspended wieder in Status Started zu versetzen.
- Neue Werte in der OpcServerState Enumeration: Suspending und Suspended.
- Erweiterung der OpcSession.Close-Methode, um nicht länger nur Diagnose-Daten zu verwerfen. Anstelle dessen wird jetzt auch wirklich die Sitzung geschlossen.
- Fehlende Code Dokumentation für Subscription-bezogene Typen im Client Namensraum.

VERÄNDERT

- Die Methode OpcSession.Close schließt ab sofort die Sitzung und invalidiert den Session-bezogenen Identifier.
- OpcClient Methoden validieren ab sofort den Verbindungs-Status mit einer OpcException (mit BadServerNotConnected) anstatt einer InvalidOperationException.
- Der Standardwert der QueueSize-Eigenschaft der OpcMonitoredItem-Instanz, welche über OpcClient.SubscribeEvent(s) erstellt wird, wurde von int.MaxValue auf uint.MaxValue geändert, damit der Server als maximale Größe der Queue für Event-Benachrichtigungen die maximal unterstützte verwendet (was ursprünglich auch das mit dem falschen Wert int.MaxValue das verfolgte Ziel war).
- Die Eigenschaft OpcMonitoredItemStatus.DataItemQueueSize wurde in QueueSize umbenannt, um der Benennung der korrespondierenden Eigenschaft in der OpcMonitoredItem-Klasse zu entsprechen (welche die angeforderte Größe während die erstere die revidierte Größe repräsentiert).
- Der Rückgabetyp der OpcMonitoredItemStatus.Id-Eigenschaft wurde von Int32 in Int64 geändert, damit dieser entsprechend dem zugrundeliegenden nicht-CLS konformen UInt32 des Stacks der Foundation repräsentieren kann.
- Die letzten API-Änderungen der OpcMonitoredItem wurden auf die OpcMonitoredItemStatus übernommen und somit die Eigenschaft OpcMonitoredItemStatus.AutoFlushCache in QueueMode umbenannt (wie es in der OpcMonitoredItem Klasse der Fall ist).

BEHOBEN

- Es wurde der falsche Status Code für Kommunikationsfehler verwendet anstelle von BadNotConnected wird nun BadCommunicationError genutzt.
- Der OpcClient verbindete sich zu einem (möglicherweise unsicheren) Fallback Endpunkt im Falle einer vordefinierten SecurityPolicy, die vom Server nicht unterstützt wird.
- Der Wert der OpcSubscription.TimestampsToReturn-Eigenschaft wurde nicht auf die MonitoredItems der Subscription übernommen, wenn diese zuvor nicht zusätzlich geändert wurden.
- Schreibfehler in der Code Dokumentation der RegisterNode-Methoden.

2.10.0.3 (freigegeben am 11.09.2020)

BEHOBEN

- Problem beim Verbinden zu OPC Classic Servern.

2.10.0.2 (freigegeben am 09.09.2020)

BEHOBEN

- Problem beim Vergleichen von OpcNodeIds, wenn diese ein Byte-Array als Wert der Node ID verwenden.

2.10.0.1 (freigegeben am 15.07.2020)

NEU

- NuGet Paket Beschreibung wurde aktualisiert.

2.10.0.0 (freigegeben am 14.07.2020)

NEU

- Aktivierung der automatischen Initialisierung von Nodes, welche über den AddNodes Dienst hinzugefügt werden. Das Ergebnis entspricht dem Verhalten das beim Hinzufügen von Nodes in einem benutzerdefinierten NodeManager Anwendung findet.
- Erweiterung der OpcCertificateManager.SaveCertificate Methode zur Ermittlung des Formats des Zertifikats anhand der verwendeten Dateierweiterung im angegebenen filePath Parameter.
- Einführung der OpcValueRange.Of Methoden, um dynamisch die zugehörige OpcValueRange für einen spezifischen Datentypen zu ermitteln.
- Nicht-generische OpcAnalogItemNodes können jetzt auch selbstständig die entsprechenden OpcValueRanges für die InstrumentRange und EngineeringUnitRange festlegen (über das DataType Attribut).
- Unterstützung zur Kodierung/Dekodierung von System.Decimal (nicht unterstützt durch OPC UA v1.03) mittels System.Double. Dies trifft auf Felder von Datenstrukturen zu.
- ProductUri-Eigenschaft in OpcClient und OpcServer zur Vereinfachung der Konfiguration dieser.
- Automatische Initialisierung der ProductUri-Information in Client- und Server-Anwendungen.
- Einführung der Verwendung/Unterstützung des aktuell gültigen SynchronizationContext's, um Benachrichtigungen im selben Kontext zu verarbeiten, unter dem die zugehörige Subscription erstellt wurde.
- Verbesserungen der Entwicklererfahrung in GUI Anwendungen (wie Windows Forms) bezüglich dem korrekten Multi-Threading in GUI Anwendungen bei der Behandlung von empfangenen Benachrichtungen.
- Einführung von OpcNode.UpdateChanges Methoden zur Benachrichtigung über Änderungen im Bezug einer Node (und deren Kind-Nodes).
- Fehlende Dokumentation wurde auf verschiedenen Typen ergänzt.

VERÄNDERT

- OpcNode.OnAfterApplyChanges und OpcNode.OnBeforeApplyChanges nutzen jetzt OpcNodeChangesEventArgs anstelle von EventArgs.
- OpcNode.AfterApplyChanges und OpcNode.BeforeApplyChanges übergeben jetzt OpcNodeChangesEventArgs anstelle von EventArgs an Eventhandler.

BEHOBEN

- Problem beim Schreiben von AnalogItemNodes, welche über einen benutzerdefinierten NodeManager ohne InstrumentalRange bereitgestellt werden, konnten beim Schreiben keine Validierung des Wertes durchführen.
- Problem mit nur einmal aktualisierten SourceTimestamp in OpcValues (wenn nicht explizit ein SourceTimestamp angegeben wurde) beim Schreiben von Variablen.
- Problem im OpcNodeHistorian (bei der Zuweisung eines solchen) beim Import von Nodes aus einem NodeSet mit aktiviertem Historizing-Attribute während zugleich die Kindknoten HistoryConfiguration und/oder AggregateConfiguration fehlten.
- Problem beim Linken der Assembly in C++ CLR Projekten mit C++/CLI, welcher den Fehler C2686 auslöst.
- Problem mit NullReferenceException's unter .NET Core 3.1 beim Empfang von Event-Benachrichtigungen.
- Problem mit nicht aufgelösten NodeIds, wenn der NamespaceIndex Eins verwendet wird (welcher für den Core/Application Namensraum reserviert ist).

2.9.2.1 (freigegeben am 08.05.2020)

BEHOBEN

- Problem beim Lesen von strukturierten Datentypen, welche ein Feld eines Arrays enthalten unter Verwendung von OpcClient.UseDynamic gleich true.

2.9.2.0 (freigegeben am 06.05.2020)

NEU

- Weitere Überladungen der Konstruktoren der OpcArgument-Klasse, um beliebige OpcDataTypes, den Identifier eines Datentypen und benutzerdefinierte Array-Dimensionen festzulegen.
- Unterstützung von n-dimensionalen (= multi-dimensionalen) Arrays von strukturierten Datentypen.
- Unterstützung von n-dimensionalen Variablen-Nodes (einschließlich neuer ArrayDimensions-Eigenschaft in der OpcVariableNode-Klasse).
- Unterstützung von null Referenzen als Feldwert für Felder die einen Referenz-Datentypen als Feldwert verwenden.
- Unterstützung der NodeClass ObjectType im AddNodes-Dienst.
- Unterstützung der NodeClass VariableType im AddNodes-Dienst.

BEHOBEN

- Problem bei der automatischen Definition des ValueRanks in Falle von n-dimensionalen benutzerdefinierten strukturierten Datentypen.
- Problem im Stack bei der Unterstützung von n-dimensionalen benutzerdefinierten strukturierten Datentypen.
- Probleme beim Laden von UANodeSets mit erweiterten Werten für die Attribtue AccessLevel und UserAccessLevel in UAVariable-Einträgen (bezieht sich auf den Stack v1.04).
- Problem mit NullReferenceException's beim Abruf von OpcDataTypeInfo.BaseType in Falle von in NodeSets definierten Datentypen, welche UATypeDefinitions ohne Felder verwenden.
- Problem mit NullReferenceException's bei der Definition von Encoding-unabhängigen Datentypen (= Enumerationen) als Teil des Adressraumes. Dieses Problem betrifft nur die Server-API.
- Problem mit falsch bestimmten ValueRank im Fall von System.Byte[], der im Stack der Foundation als ByteString interpretiert wurde.

2.9.1.0 (freigegeben am 22.04.2020)

BEHOBEN

- Problem, das zu doppelt durchsuchten Referenzen zwischen übergeordneten/untergeordneten Knoten führte, die mit einem NodeSet importiert wurden.
- Probleme mit neueren Versionen von System.ServiceModel.Primitives. Rollback der Abhängigkeit zu v4.5.3 in allen Zielframeworks.
- Probleme mit nachfolgenden Evaluierungslizenzen, die für die Bundle-Lizenzierung verwendet werden (Client + Server).
- Problem mit falsch bestimmten ValueRank im Fall von System.Byte[], der im Stack der Foundation als ByteString interpretiert wurde.
- Problem mit dem Ergebnis von OpcNamespace.Get(int, Uri), das zu einem OpcNamespace.Value führt, der den durch den Parameter namespaceUri definierten Uri.OriginalString nicht ausdrückt.
- Problem mit kaskadierten benutzerdefinierten Datentypen, wenn sie im DataTypeEncoding kein benutzerdefinierten NamespaceUri festlegen. Dieses Problem betrifft nur die Server-API.
- Problem mit nicht angewendetem BrowseName in AddNodes bei Nodes der NodeClasses die NodeTypen definieren. Dieses Problem betrifft nur die Server-API.
- Probleme mit Nodes der NodeClasses die NodeTypen definieren, welche beim AddNode falsch überprüft wurden. Dieses Problem betrifft nur die Server-API.

2.9.0.0 (freigegeben am 01.04.2020)

NEU

- Vereinheitlichte Produktlizenzierung, welche sich nicht länger vom verwendeten Framework unterscheidet.
- Alle Licenser Klassen stellen ab sofort diverse FailIf...- und ThrowIf...-Methoden bereit, um bei Bedarf im benutzerdefinierten Code die verwendete Lizenz zu verifizieren. Verwenden Sie FailIf...Methoden in ihrer DEBUG Konfiguration und ThrowIf-Methoden in ihrer RELEASE Konfiguration, um sich vor der nicht-lizenzierten Verwendung des SDKs zu schützen.
- Neue GetNodeType(...) Methoden in OpcClient (nur for Event-Node-Typen).
- Neue GetNodeTypeSystem(...) Methoden in OpcClient (nur für Event-Node-Typen).
- Neue GetNodeType(...) Methoden in OpcNodeSet.
- Neue GetNodeTypeSystem(...) Methode in OpcNodeSet.
- OpcMethodNode unterstützt benutzerdefinierte Eingangs- und Ausgangsparameter, wenn die Schnittstelle IOpcMethodCommand verwendet wird.
- Neue GetNodeType(...) Methoden und GetNodeTypeSystem() Methode in OpcNodeSet.
- OpcNode vererbt ab sofort seinen OpcNode.Namespace an seine Kind-Nodes (falls diese Standard-Namespace bezüglich ihrer OpcNode.Id verwenden).
- Neue Spezialisierungen der OpcTypeNodeInfo: OpcObjectTypeNodeInfo und OpcVariableTypeNodeInfo.
- Verbesserte Erfahrung beim Debugging und Inspizieren der Eigenschaften von OpcNodeInfo Instancen (und deren Unterklassen).
- Browse unterstützt ab sofort das zusätzliche Lesen von weiteren Attributen, neben den Attribute die bereits durch den Browse-Dienst unterstützt werden.
- OpcNodeInfo implementiert jetzt IOpcNodeInfo, um eine flüssigere Verwendung von Node-Instanzen unabhängig von ihren Ursprung zu ermöglichen.
- Weitere konfiguierbare Dienste in OpcClientServices: Browse und ReadNodes.
- Erheblich verbesserte Leistung beim Erstellen von Eventfiltern.
- IOpcNodeInfo definiert nun weitere Methoden: AttributeValue(...), Child(...) und Children().

- Unbeschränkte Browse-, Read und Write-Operationen aufgrund automatischer Partitionierung von Anfragen (OpcClient).
- OpcClient.CertificateJudgement-Ereignis zur Behandlung von Problemen in Zusammenhang mit Zertifikaten, nachdem eine Verbindung hergestellt und Zertifikats-Informationen bereits erfolgreich validiert wurden.
- OpcModel zur Repräsentation von Model-Informationen in NodeSets.
- OpcMemberSwitch.Conditions-Eigenschaft zur Information über die Bedingungen unter denen ein Switch ausgewertet wird.
- Direkte Unterstützung von .NET Core 3.1 als eines unserer Zielframeworks.
- ReplaceNode-Methode in OpcNodeManager, um existierende Nodes rekursiv auch bezüglich ihrer Notifiers und MonitoredItems zu ersetzen.
- OpcNodeSet wurden zusätzliche Metadateneigenschaften (LastModified und Name) hinzugefügt.
- Eine allgemeine Implementierung der IOpcMethodCommand-Schnittstelle unter Verwendung von Delegaten: OpcMethodDelegateCommand.
- Fehlende Dokumentation für Collections und einigen anderen Typen hinzugefügt.
- OpcNodeSetManager.ImplementNodeCallback zum Implementieren von Nodes mithilfe eines einfachen Callbacks.
- OpcData.GetNodeId unterstützt jetzt auch Array-Typen.
- Codierungsinformationen im Adressraum des Servers (intern) hinzugefügt.
- Generische Implementierung von IOpcMethodCommand für Delegaten OpcMethodDelegateCommand.
- Unterstützung von System.Object und System.Object[] zum Kodieren / Dekodieren von Datentypen.
- Unterstützung von Serialisierungs- / Deserialisierungsattributen, die in System.Runtime.Serialization für Datentypen definiert sind.
- OpcDataTypeMembersAttribute zum Filtern von Elementen, die für Datentypen codiert / decodiert werden sollen.
- OpcDictionary<TKey, TValue> als einfacher und nicht standardmäßig verfügbarer Datentyp zum Transportieren von Wörterbüchern mit OPC UA.
- Es ist nicht länger notwendig einem Namen noch einem Identifier einer benutzerdefinierten Wurzel-Node, welche in OpcNodeManager.CreateNodes(...) zurückgegeben wird, manuell den Standard-Namespace des benutzerdefinierten NodeManagers zuzuweisen.
- Beachtung der Reihenfolge der Felder wie diese innerhalb ihrer Vererbungshierarchie deklariert sind, um immer zuerst die Felder des Basisdatentyps zu definieren.
- Neue GetDataType(...) Methoden in der OpcClient Klasse.
- Neue GetDataType(...) Methoden in der OpcNodeSet Klasse.
- IOpcNodeInfo definiert nun die Methoden Child(...) und Children(), um die Kind-Nodes von Instanz-Nodes abzurufen.
- Neue OpcNamespaceCollection Klasse.
- OpcNodeSet.Namespaces Eigenschaft um die Namespaces bereitzustellen, auf die im Nodeset verwiesen wird.
- IOpcMethodCommand zur Definition/Verwendung von benutzerdefinierten Command-Instanzen anstelle der Definition eines expliziten Delegaten.
- OpcMethodCommands Klasse, welche vordefinierte Commands wie den NotImplemented-Command bereitstellt.
- Erweiterung der OpcMethodNode zur Unterstützung von IOpcMethodCommand Instanzen.
- Erweiterung der generischen OpcDataTypeNode<T> Klasse zur Unterstützung von Klassen- und Struktur-Typen, welche wiederum auf vorhandene OPC UA Strukturen abgebildet werden.
- Neue nicht-generische OpcDataTypeNode Klasse.
- Statische OpcContext.Empty Eigenschaft zur Definition einer „leeren“ Kontext-Instanz.

- IOpcNamespaceResolver wurde in der OpcContext Klasse implementiert und ermöglicht somit das Auflösen von Namespaces von OpcNodeId-Instanzen.
- OpcTypeSystem Klasse, welche als Basis für die existierende OpcDataTypeSystem und die neue OpcNodeTypeSystem Klasse dient.
- OpcDataTypeDictionary.Documentation Eigenschaft.
- OpcNodeSetManager Klasse, welche das Einfache Importierten/Hosten von OpcNodeSets mittels OpcNodeSetManager.Create(...) unterstützt.
- Protected virtual Methoden OpcNodeManager.ImportNodes() und OpcNodeManager.ImplementNode(IOpcNode).
- Vereinheitlichung / Vereinfachung der Auflösung des DataType-Attributes insbesondere bei der Verwendung von nicht-built-in Datentypen bei der Definition von Variablen- und Property-Nodes. Dies betrifft die folgenden Node-Klassen: OpcDataItemNode, OpcDataItemNode<T>, OpcAnalogItemNode, OpcAnalogItemNode<T>, OpcDataVariableNode, OpcDataVariableNode<T>, OpcConditionVariableNode, OpcConditionVariableNode<T>, OpcPropertyNode, OpcPropertyNode<T> and OpcVariableNode.

VERÄNDERT

- **Lizenzschlüssel die vor der Version 2.9 ausgestellt wurden funktionieren nicht länger. Unsere Kunden werden entsprechend neue Lizenzschlüssel erhalten die dann auch in allen Frameworks funktionieren werden.**
- OpcAttributeWriteAccess werden jetzt UInt32 als Basistyp.
- Der Standardwert der OpcAutomatism.AlwaysParseStringIdentifiers-Eigenschaft ist jetzt false. Hierdurch wird beim Aufruf des Konstruktors `OpcNodeId(string)` der Wert nicht länger automatisch geparst. Für weitere Details siehe OpcAutomatism.AlwaysParseStringIdentifiers
- OpcReadOnlyNamespaceCollection wurde vom Namespace Opc.UaFx.Server in den Namespace Opc.UaFx verschoben.
- Standardwert des DataType-Attributs von OpcDataTypeDictionaryNode von Byte auf ByteString, um der Definition der Spezifikation zu entsprechen.
- Description Eigenschaft wurde auf Documentation umbenannt; um die Konformität zur Spezifikation sicherzustellen. Betroffen sind: IOpcDataTypeInfo, OpcDataTypeInfo und OpcDataFieldInfo to Documentation.
- OpcEncoding.All(...) erzeugt nun die OpcEncoding.Id aufgrund des Standard-BrowseNames für die verschiedenen Encoding-Types.
- OpcNamespace.ToString() liefert ab sofort einen string, welcher ausschließlich den OpcNamespace.Value + OpcNamespace.Index (falls nicht null) enthält.
- OpcNodeSet.GetDataTypeSystem() liefert nun eine OpcDataTypeSystem Instanz, welche den typischen BrowseName der jeweiligen Art der Codierung (welche vom System repräsentiert wird).
- IOpcTypeProvider.GetType(OpcEncoding) wurde entfernt, weil Codierungsinformationen nur auf Datentypen zutreffen.
- OpcTypeCategory.EventType und OpcTypeCategory.ReferenceType wurden entfernt.

BEHOBEN

- Probleme bei der Validierung veralteter Zertifikate oder Zertifikate, die gegen bestimmte Richtlinien für allgemein gültige Zertifikate verstoßen. Solche Zertifikate können jetzt vom CertificateValidationFailed-Ereignis verarbeitet werden.
- Problem bei der automatischen Anwendungskonfiguration im Falle, dass die Entry-Assembly entweder kein AssemblyTitleAttribute definiert oder dieses string.Empty als Assembly-Title verwendet.
- Problem bei Änderungen von Variable-Node-Werten beim Wechsel zwischen verschiedenen

komplexen Datentypen (die über ein ExtensionObject bereitgestellt werden).
- Falsch erzeugte EncodingMask im Falle von mehr Datentypen mit mehr als einen optionalen Feld.
- Falsche Interpretation von OpcArgument.IsArray und OpcArgumentInfo.IsArray.
- Ein initial festgelegter OpcVariableNode.Timestamp wird nicht länger während der Erstellung des Addressraumes verworfen.
- Problem mit falsch ermittelten Standardwerten/-rängen und Standard-Datentypen-Identifiern, durch OpcDataTypes.GetValueRank(..) und OpcDataTypes.GetNodeId(...).
- Probleme mit DataTypeDictionary-Nodes, welche keine DataTypeDescription-Nodes definieren, während der Erzeugung des DataTypeSystems beim Aufruf von OpcClient.GetDataTypeSystem().
- Probleme bei der Dekodierung von strukturierten Daten, welche den neuen Typen Opc.UaFx.Bit verwenden.
- Probleme bei der Verwendung von OpcNodeId.UriIdentifier(...), wenn der Wert von OpcNodeId.Value kein System.String-Wert ist.
- OpcNamespace behält nun den Original-String bei, der beim Definieren einer OpcNamespace-Instanz verwendet wurde. Jegliche darauffolgende Änderungsmäßnahmen (durch die System.Uri-Klasse) werden jetzt vermieden, sodass die OpcNamespace.Value Eigenschaft immer den gleichen System.String-Wert liefert wie bei der Definition der Instanz verwendet.
- Probleme die in manchen Situationen dazu führten, dass eine neu erstellte OpcNode-Instanz zu einen späteren Zeitpunkt nicht mehr umbenannt werden konnte, wenn bei der Erstellung dieser kein Name angegeben wurde.

2.8.3.1 (freigegeben am 24.01.2020)

BEHOBEN

- Problem mit nicht mehr funktionierenden Subscription-Handlern nach erfolgreichem erneutem Verbindungsaufbau.
- Problem mit verworfener Socket-Instanz in verworfenen TcpMessageSocket-Instanzen.

2.8.3.0 (freigegeben am 16.01.2020)

NEU

- Unterstützung von OpcExceptions in Delegaten welche für Methoden-Nodes verwendet werden. Das ermöglicht die Definition eines Bad-Ergebnisses, ohne dieses manuell über den OpcMethodContext-Parameter setzen zu müssen.
- Unterstützung für weitere Access Levels (StatusWrite und TimestampWrite).
- Ergänzung der Dokumentation für OpcAccessLevel, OpcVariableNode.AccessLevel und OpcVariableNode.UserAccessLevel.
- Untersützung des Abbruchs einer Verbindung während eines ausstehenden Wiederverbindungsversuches im OpcClient.
- Änderungen um die Erfahrung mit UWP Anwendungen im Bezug auf die .NET Native Tool Chain zu verbessern.

VERÄNDERT

- Überarbeitete Logik zur Änderung der OpcClient.State-Eigenschaft um besser den Verlust einer Verbindung zum Server feststellen zu können.
- OpcClient.State-Eigenschaft ändert sich jetzt automatisch nur noch auf den Wert Disconnected (während einer laufenden Anfrage), wenn der ReconnectTimeout größer als 0 ist (Wiederverbindungslogik wird vom Anwender implementiert).

BEHOBEN

- Problem mit nicht übernommenen Status (Code) Informationen, welche an eine der Konstruktoren der OpcValue-Klasse übergeben wurden.
- Problem mit TypeDictionaries, welche keine NamespaceUri bereitstellen.

2.8.2.1 (freigegeben am 13.12.2019)

BEHOBEN

- Problem beim Veröffentlichen von parallelen Subscriptions beim Aufrufen eines Methodenknotens im OpcClient.
- Problem mit nicht aktualisiertem OpcVariableNode.Value, falls einem variablen Knoten kein benutzerdefinierter WriteVariableValueCallback zugeordnet ist.
- Problem das dazu führte, dass der Wert der OpcClientKeepAlive.Interval-Eigenschaft keine Anwendung auf die aktuelle Sitzung fand.

2.8.2.0 (freigegeben am 06.11.2019)

NEU

- Einführung von Opc.UaFx.Bit als eine .NET-basierte Repräsentation des Foundation-definierten „Bit“ Werttypen für den bisher keine konkrete Implementierung existiert.

VERÄNDERT

- Explizit kodierte Informationen bekannt als „EncodingMask“ eines strukturierten Datentypen werden jetzt über den neu eingeführten Opc.UaFx.Bit Typen anstelle von System.Boolean abgebildet.

BEHOBEN

- Problem mit gemischt-strukturierten Datentypen welche führende Boolean-Werte gefolgt von weiteren Nicht-Boolean-Werten verwenden ergaben einen fälschlich kodierten Datenstrom.
- Problem beim Verbindungsaufbau zu einen OPC Classic Server in der OPC Watch, wenn kein benutzerdefiniertes Zertifikat verwendet wird.
- Problem in OpcServerInfo.FromProcess(...) im Falle, dass die ProductVersion der FileVersionInfo der ProcessModule-Instanz von Process.MainModule einen Version-String bereitstellt, der nicht ausschließlich aus Zahlen und Punkten besteht wie z.B. '1.2.3.4-preview'.

2.8.1.3 (freigegeben am 24.10.2019)

BEHOBEN

- Problem beim Austausch von strukturierten Datentypen ohne die Verwendung der anstehenden Metadaten-Unterstützung.

2.8.1.2 (freigegeben am 23.10.2019)

VERÄNDERT

- iconUrl und licenseUrl Informationen in NuGet Paketen aufgrund ihrer Abkündigung.

BEHOBEN

- Problem beim Verbinden zu OPC Classic Servern (An internal error occurred as a result of a

programming or configuration error).
- Problem der falschen Interpretation von Int32 Werten als Rückgabetyp von Methodenknoten.

2.8.1.1 (freigegeben am 11.10.2019)

NEU

- Unterstützung für Methodenparameter mit „System.Object“ implementiert, die als OPC UA Variantenwerte interpretiert werden.

BEHOBEN

- Probleme mit nicht berücksichtigten Methodenparametern vom Typ „System.Object“.
- Problem beim Lesen großer Dateiknoten (die Dateien darstellen, deren Inhalt die MaxByteStringLength überschreitet).

2.8.1.0 (freigegeben am 25.09.2019)

NEU

- RegisterNodes und UnregisterNodes Methoden auf der OpcClient-Klasse zur Registrierung von Nodes für die ein OPC UA Server für die aktuelle Sitzung einen schnelleren Zugriff realisieren sollen.
- Unterstützung der OPC Classic API in .NET Standard, damit die API auch in .NET Core Anwendungen unter Windows zur Verfügung steht.
- Neue Methoden OpcDataObject.HasField(name) und OpcDataObject.TryGetField(name, out field).
- Verbesserte Performance durch den Ausschluss von Foundation-definierten Data Type Dictionaries beim Abrufen des Datentypen-Systems eines Servers.
- Neue Eigenschaften OpcClient.UseDynamic und geänderter Standardwert der OpcAutomatism.UseDynamic-Eigenschaft (auf den Wert false) um die Performance beim Verbindungsaufbau zu steigern, wenn keine Information über Datentypen benötigt werden.
- Unterstützung von Servern, beim Abrufen der Data Type Dictionaries, die kein Bereichs-basiertes Lesen von Daten unterstützen.

BEHOBEN

- Absturz durch eine nicht behandelte Ausnahme beim Aufruf von Socket.Shutdown().

VERÄNDERT

- Die Eigenschaft OpcAutomatism.UseDynamic besitzt ab sofort den Standardwert false. Die Änderung dieser oder OpcClient.UseDynamic-Eigenschaft auch den Wert true ist notwendig damit das OpcDataObject genutzt werden kann.

2.8.0.0 (freigegeben am 18.09.2019)

NEU

- Einführung einer Reihe neuer Klassen zur Unterstützung der Definition, Verwendung und Reflektion benutzerdefinierter Datentypen in Clientanwendungen.
- Neue OpcNodeSet-Klasse, um eine Datei-, Zeichenfolgen- oder Stream-basierte Quelle bereitzustellen, die Informationen zum Datentypensystem aus einem UANodeSet bereitstellt.
- Die neue OpcClient.NodeSet-Eigenschaft wurde implementiert, um eine Instanz der OpcNodeSet-Klasse festzulegen, von der Datentypensysteminformationen abgefragt werden sollen, anstatt das Datentypensystem des Servers abzufragen.

- OpcClient.GetDataTypeSystem()-Methode, zum Ermitteln des Datentypsystems das vom Client zum Auflösen von Datentypinformationen verwendet wird.
- OpcAutomatism.UseDynamicTypeRegistration-Eigenschaft zur Steuerung der automatischen Typreflexion, mit der Client- und Serveranwendungen die verschiedenen deklarierten Daten- und Ereignistypen bestimmen.
- Einführung der neuen OpcAutomatism.UseDynamic-Eigenschaft zur Steuerung der dynamischen Typauflösung bei Verwendung von benutzerdefinierten, aber nicht explizit deklarierten Datentypen in Clientanwendungen.
- Neue OpcNodeValue<T>-Klasse, OpcAttributeValue<T> und IOpcNodeAttribute mit IOpcNodeDescriptor in Verbindung mit OpcNodeValuePair eingeführt, um „knotengebundene“ Werte zu deklarieren.
- Erweiterung der OpcClient-Klasse mit neuen ReadNodeValue- und WriteNodeValue-Methoden zum Lesen und Schreiben von „knotengebundenen“ Werten.
- Verbesserte Ausnahmebehandlung, um StackTrace-Informationen vom Benutzercode bis zur Foundation- und Kommunikationsebene beizubehalten.
- Implementierung der neuen OpcClient.KeepAlive-Eigenschaft samt entsprechender OpcClientKeepAlive-Klasse, um die Keep-Alive-Aktionen in einer Clientanwendung zu steuern und zu überwachen.
- Implementierung der dynamischen Typkonvertierung in OpcValue.As<T>()
- Neue AsValue<T>()-Methode in der OpcValue Klasse, um eine generische Definition des Werts mithilfe der neuen OpcValue<T>-Klasse bereitzustellen.
- OpcNamespace stellt jetzt die Namespace-Zeichenfolge / Uri mit der neuen OpcNamespace.Value-Eigenschaft bereit.
- Die Standardtransportlimits wurden bei DefaultMaxByteStringLength von 64 KB auf 5 MB und bei DefaultMaxMessageSize von 1 MB auf 10 MB erhöht.
- Die inline behandelten Ausnahmen bei Verwendung der XmlSerializer Klasse beim Start wurden behoben.
- Verbesserte Ausnahmebehandlung und -verfolgung beim Kodieren und Dekodieren von Daten (siehe OpcTypes.EncodingFailed und OpcTypes.DecodingFailed).
- Unterstützung zum Tracen aller Kodierungsoperationen mit Hilfe der neuen Klassen OpcEncodingStackFrame und OpcEncodingStackTrace.
- Einführung der Unterstützung des Tracings der Decodierung und Codierung mithilfe der neuen Klassen OpcEncodingStackFrame und OpcEncodingStackTrace.
- „int ResolveIndex (string namespaceValue)“ und „string ResolveValue (int namespaceIndex)“ zu IOpcNamespaceResolver hinzugefügt.
- OpcClient bietet jetzt Methoden, um Aufrufe von CallMethod mithilfe der neuen generischen Methoden CallAction und CallFunc zu vereinfachen.
- Zusätzliche Überladungen zu OpcBrowseNode hinzugefügt, um OpcNodeIds zu unterstützen, die auf bestimmte Verweistypen verweisen.
- OpcNodeInfo bietet jetzt zusätzliche Überladungen von Children(…) und Parents(…), die einen OpcReferenceType-Wert akzeptieren.
- OpcResult liefert jetzt mit GetBaseResult() das Root-Ergebnis.
- OpcEventNode.EventTypeId kann jetzt mit dem hinzugefügten Setter beschrieben werden.
- Einführung von OpcEventTypeAttribute zur Client-seitigen Definition benutzerdefinierter Ereignistypen, die zur Verarbeitung von Ereignisbenachrichtigungsdaten in Clientanwendungen verwendet werden.
- Falsch verwendeter Statuscode beim Dekodieren im Foundation Stack korrigiert (vorher: BadEncodingError, jetzt: BadDecodingError).
- Fehlende SourceType-Eigenschaft zu Opc.Ua.Schema.Binary.FieldType hinzugefügt.

- Fehlende BaseType-Eigenschaft zu Opc.Ua.Schema.Binary.BaseType hinzugefügt.

BEHOBEN

- Falsche verwendete Variable beim Konfigurieren von Opc.UaFx.Client.OpcSubscription.LifetimeCount.
- Probleme bei der Serialisierung von OpcException-Objekten.

VERÄNDERT

- OpcDataTypeAttribute ist jetzt auf Klassen und Strukturen anwendbar.
- OpcNodeManager.NamespaceIndexes und OpcNodeManager.NamespaceUris wurden in der neuen OpcNodeManager.Namespaces-Eigenschaft zusammengefasst.
- IOpcReadOnlyDataStore wurde in IOpcReadOnlyNodeDataStore umbenannt.
- IOpcNamespaceResolver „Uri Resolve(int namespaceIndex)“ und „int Resolve(Uri namespaceUri)“ sind jetzt veraltet.

2.7.5.1 (freigegeben am 15.08.2019)

NEU

- Verwendung der OpcNominalNodeIdFactory (über OpcNodeId.Factory) zur Auflösung von OPC Classic Tag-Namen unter Verwendung des OPC Classic Server spezifischen Tag-Separators.

BEHOBEN

- Problem mit mehrfach eingetragenen Serialisierungsdaten von Ausnahmen, welche zu SerializationExceptions an Stelle der eigentlichen Ausnahmen führte.

2.7.5.0 (freigegeben am 13.08.2019)

NEU

- Verbesserte Dokumentation der API: IOpcNode, IOpcNodeInfo and OpcNode.
- Änderungen am Stack um sicherzustellen, dass Node-Manager Locks solange nicht freigegeben werden, solange ein Monitored-Item angelegt, modifiziert oder gelöscht wird.
- System.ServiceModel.Primitives wurde von Version 4.5.x auf Version 4.5.3 aktualisiert, weil in Version 4.5.x ein Fehler das korrekte Kopieren der „System.Private.ServiceModel.dll“ in .NET Core Anwendungen beim Publish für Windows (win-xxx) verhinderte.

BEHOBEN

- Probleme beim Kompilieren mittels .NET Native tool chain.
- Probleme mit mehrfach auftretenden Benachrichtigungen nach einer erneuten Verbindung zum Server.
- Probleme mit inline geparsten OpcNodeId's beim Erstellen einer OpcNodeId mit einem String-Identifier.
- Problem mit falsch verwendeter Basis-Adresse und Security-Poilicies im OpcDiscoveryServer.
- Probleme mit unnötigerweise wiedererstellten OpcMonitoredItem Instanz welches dazu führte, dass das Vergleichen dieser nicht länger funktionierte.

VERÄNDERT

- OpcMonitoredItem.AutoFlushCache wurde durch die neue Eigenschaft OpcMonitoredItem.QueueMode ersetzt, um den verwendeten Mechanismus beim Überlauf der

Benachrichtigungsqueue klarer auszudrücken.

2.7.4.0 (freigegeben am 07.06.2019)

NEU

- Die Domänen-Namensauflösung, welche während der Validierung von Client- / Server-Zertifikaten verwendet wird, verwendet jetzt alternative Mechanismen zur Auflösung, falls kein DNS Server im Betriebssystem konfiguriert wurde.
- Die im Falle von HTTPS (im Stack der Foundation) nicht länger verwendete OperationTimeout findet wieder Anwendung.
- Neue Ereignisse in der OpcNodeManager Klasse: MonitoredItemsCreated, MonitoredItemsModified und MonitoredItemsDeleted.

BEHOBEN

- SocketExceptions unter macOS während der Validierung von Domänen-Namen in Zertifikaten, wenn kein (erreichbarer) DNS Server im Betriebssystem eingerichtet wurde.

VERÄNDERT

- Vereinzelte Eigenschaften in den Klassen OpcMonitoredItem, OpcServiceCounter, OpcSessionDiagnostics und OpcServerDiagnostics liefern anstelle von int- jetzt long-basierte Werte um weiterhin die CLS Compliance zu wahren, obwohl der Stack der Foundation unsigned Datentypen verwendet.

2.7.3.1 (freigegeben am 23.05.2019)

NEU

- [Client SDK] Unterstützung des .NET Frameworks 4.6.

BEHOBEN

- Problem beim Verbinden zu Server, welche als Protokoll HTTPS auf macOS und Linux verwenden. Das Problem äußerte sich in der OpcException: An unrecognized response was received from the server.

2.7.3.0 (freigegeben am 17.05.2019)

NEU

- Die lokale X509Certificate2 Instanziierung wurde durch das OpcCertificateManager.QueryInstance event zum Laden von Zertifikat-Informationen ersetzt. Dies ermöglicht die benutzerdefinierte X509Certificate2 Instanzenerzeugung unter Verwendung eines entsprechenden EventHandlers.
- Die neue OpcCertificateStores.PathProbings Eigenschaft ermöglich die benutzerdefinierte Anpassung der Probing Mechanismen die zur Ermittlung der standardmäßig verwendeten Speicherorte der Zertifikatspeicher verwendet werden.
- Verbesserung der Benutzererfahrung bezüglich verschiedener (kleiner) Probleme in Subscriptions und MonitoredItems (insbesondere beim automatischen Neuverbinden des Clients).
- Weitere AddMonitoredItem(...) Überladungen wurden der OpcSubscription Klasse hinzugefügt.

BEHOBEN

- Problem beim Verbinden zu Server, welche als Protokoll HTTPS verwenden. Das Problem äußerte

sich in der OpcException: An unrecognized response was received from the server.

VERÄNDERT

- Die Implementierung der OpcMonitoredItem und OpcSubscription Klasse wurde aktualisiert: Hierbei wurden zum Teil „int“-basierte Eigenschafen auf „long“ umgestellt, um die zugrundeliegenden „uint“ Eigenschaften des Foundation Stacks entsprechend CLSCompliant anzubieten.

2.7.2.0 (freigegeben am 10.05.2019)

NEU

- Unterstützung von TypeDefinition Informationen beim Browsen von Instanz-Nodes durch die neue OpcInstanceNodeInfo Klasse.
- Neue Socket Eigenschaft auf der OpcSession Klasse zur Identifizierung der Socket Identitätsinformationen die für den Lokalen und den Remote Endpunkt verwendet werden.
- Produktsymbol der NuGet Pakete wurde geändert.
- [Client SDK] Aktive Unterstützung von IL2CPP in Unity Projekten. Hierdurch werden vereinzelte Überlaufausnahmen durch IL2CPP gelöst.

BEHOBEN

- Probleme beim gleichzeitigen Browsen von mehr als einen Referenzentypen, welche zu einer ArgumentOutOfRangeException führten.
- Probleme die manchmal zu veralteten Subscription und Monitored Item Informationen führen konnten, wenn durch den Client eine verlorene Verbindung zum Server automatisch wiederhergestellt wurde.
- Problem mit manchmal fehlenden Client- und Subscription-basierten Notification Ereignissen - wie z.B. NotificationReceived blieb auf dem OpcClient aus.

2.7.1.0 (freigegeben am 25.03.2019)

NEU

- Verweigerung von angeforderten Node Identifiers die einen unbekannten Namespace Index Wert verwenden (im AddNodes Service, BadNodeIdInvalid).
- Verwendung des Node (Identifier) spezifischen Node Managers beim Anlegen von Nodes durch den Client, um sicherzustellen, dass ein Node nicht zu den Nodes des Node Managers des Elternknotens hinzugefügt wird, wenn dieser auf einen Namespace eines anderen Node Managers verweist.
- Node Zugriff- und Schreibe-Flanken werden ab sofort rekursiv vergeben, wenn ein Node über durch einen Client angelegt wird.

BEHOBEN

- Problem der Verweigerung von durch den Client vorgegebenen Node Identifier (AddNodes Service).
- Probleme mit benutzerdefinierten Node-Attributen zum Festlegen von Zugriff- und Schreib-Flanken beim Anlegen von Nodes durch den Client.
- Korrektur der wenig schön generierten Namespace Uri des Standard-Namespaces (unnötige '/' wurden entfernt).
- Problem mit falsch verpackten .NET Array Types bei der Verwendung von OpcValue Instanzen.
- Problem mit falsch verpackten UA FX Array Typen bei der Verwendung von OpcValue Instanzen.
- Problem mit fehlenden DataType Informationen im Falle von OpcVariableNode (nicht generischen) Instanzen, wenn diese mit einem Initialwert über den Konstruktor initialisiert werden und

anschließend ihre DataType Eigenschaft auf den entsprechenden Typen festgelegt wurde. Resultat: Die DataType Eigenschaft übernahm den neuen Wert nicht.

2.7.0.2 (freigegeben am 15.03.2019)

- **BEHOBEN:** Fehlerhafte Adressierung von nicht verwalteten Speicher (bei Verwendung von OPC Classic), welche insbesondere in Anwendungen fehlschlagen kann, wenn diese die Option LARGEADDRESSAWARE verwenden.

2.7.0.1 (freigegeben am 15.03.2019)

- **BEHOBEN:** Problem mit Gebietsschemata, welche vom OPC Classic Server zwar unterstützt werden, aber nicht vom Betriebssystem des OPC Clients.
- **BEHOBEN:** Problem bei der Kommunikation mit einem OPC Classic Server der auf einem 64 Bit System ausgeführt wird.

2.7.0.0 (freigegeben am 14.03.2019)

- **NEU:** Vollständige Überarbeitung der AddNode Unterstützung im OpcClient einschließlich einem vollständig neuen Satz von OpcAddNode command Typen wie: OpcAddFolderNode, OpcAddObjectNode, OpcAddDataVariableNode, OpcAddAnalogItemNode und einige mehr.
- **NEU:** Erstmalige vollständige Unterstützung von AddNode Service-Calls im OpcServer, welcher alle unterstützten Typ-Definitionen zum Erstellen von Nodes im Adressraum des Servers ermöglicht.
- **NEU:** Vollständige Überarbeitung der DeleteNode, AddReference und DeleteReference Unterstützung im OpcClient einschließlich der Neuauflage der OpcDeleteNode, OpcAddReference und OpcDeleteReference Commands.
- **NEU:** Überarbeitung der Unterstützung von DeleteNode, AddReference und DeleteReference Service-Calls im OpcServer unter Anwendung zusätzlicher Anfragen-Validierungen.
- **NEU:** Einführung der neuen OpcAddNodeResult Klasse, welche eine Spezialisierung der OpcResult Klasse darstellt und somit das Ergebnis, dessen Commands wie auch die NodeId nach der Ausführung eines AddNode Service-Calls bereitstellt.
- **NEU:** Der OpcNodeManager stellt ab sofort virtuelle Methoden bereit, welche beim Hinzufügen (via AddNode Service) und Entfernen (via DeleteNode Service) von Nodes zum / vom Adressraum des Managers ausgeführt werden.
- **NEU:** Verbesserte Exception Details bei der Verwendung der Services: AddNodes, DeleteNodes, AddReferences und DeleteReferences.
- **NEU:** Der Konstruktor der OpcServerIdentity Klasse stellt ab sofort eine Überladung für ein string basiertes Passwort bereit.
- **NEU:** OpcServer Ereignisse: RequestProcessing, RequestValidating, RequestValidated und RequestProcessed zur Vor- und Nachverarbeitung von Client Anfragen und den darauf gesendeten Antworten.
- **NEU:** Die OpcException stellt ab sofort Informationen über die Source, den StackTrace, Data und den HelpLink bereit (falls verfügbar) und wird im Falle einer ursprünglich anderen Exception diese nicht länger verwerfen.
- **NEU:** IOpcNode.Child(…) Methode zum Ermitteln einer IOpcNode Kind-Instanz unter Verwendung des Namens (= Browse Name) des Kindknotens.
- **NEU:** Zusätzliche OpcServerIdentity.ChangePassword(string) Methodenüberladung.
- **NEU:** Zusätzliche OpcUserNameAcl.AddEntry(…) Methodenüberladung zur Definition einer benutzerdefinierten OpcServerIdentity.

- **NEU:** Subclassing der OpcResult Klasse für benutzerdefinierte Typen.
- **VERÄNDERT:** Die OpcServerIdentity.Password Eigenschaft wurde entfernt, weil es keinen Grund gibt das intern gespeicherte Passwort nach außen weiterzugeben. Zudem ist es keine gute Praxis einen Verweis auf ein Array über eine Eigenschaft bereitzustellen.
- **VERÄNDERT:** OpcTransformUserPasswordDelegate Delegat wurde entfernt.
- **VERÄNDERT:** OpcUserNameAcl.PasswordTransform wird durch die virtuellen Methoden Matches und TransformPassword in der OpcServerIdentity ersetzt.
- **BEHOBEN:** Problem beim Schreiben von eingebauten strukturierten Werten wie OpcEngineeringUnitInfo oder OpcValueRange unter Verwendung von OpcClient.WriteNode.
- **BEHOBEN:** Problem beim Trennen einer Verbindung / Anhalten eines Clients / Servers, welches im Falle eines noch ausstehenden Verbindungsversuchs zum Absturz führen kann.
- **BEHOBEN:** Problem mit StackOverflowException's bei der Änderung der Instanz der OpcServer.Configuration Eigenschaft.

2.6.0.0 (freigegeben am 20.02.2019)

- **NEU:** OPC Classic Support via OpcClient Klasse durch Angabe der Adresse im Schema "opc.com://<host>(:<port>)/<progId>/<classId>". (nur mit .NET Framework)
- **NEU:** OpcClassicInterface, OpcClassicInterfaces, OpcClassicInterfaceCollection und OpcClassicInterfaceCategory. (nur mit .NET Framework)
- **NEU:** OpcClassicApplicationDescription, OpcClassicApplicationDescriptionCollection, OpcClassicEndpointDescription und OpcClassicEndpointDescriptionCollection. (nur mit .NET Framework)
- **NEU:** OPC Classic Discovery unter Verwendung der neuen OpcClassicDiscoveryClient Klasse, hierzu wird „OpcEnum.exe“ verwendet (eine COM Anwendung, die Teil der OPC Classic Laufzeitumgebung ist). (nur mit .NET Framework)
- **NEU:** Erweiterung der OpcDataChangeFilter Klasse zur Unterstützung eines Triggers, des DeadbandTypes und eines DeadbandValues. Unter Verwendung der Trigger Eigenschaft ist es möglich Änderungen des SourceTimestamp ebenfalls zu überwachen.
- **NEU:** Zusätzliche Überladungen der OpcClient.SubscribeDataChange Methode und zusätzliche Überladungen der OpcDataSubscribeDataChange Konstruktoren, welche einen Wert der OpcDataChangeTrigger Enumeration akzeptieren.
- **NEU:** Ein vordefinierter ApplicationName wird ab sofort als SubjectName eines Zertifikats verwendet, wenn ein neues Zertifikat (automatisch) für eine Client / Server Anwendung erstellt wird.
- **NEU:** OpcApplicationInstance stellt eine Standardimplementierung der IOpcApplicationInstance Schnittstelle bereit.
- **NEU:** OpcClient und OpcServerBase verwenden ab sofort die neue Klasse OpcApplicationInstance als gemeinsame Basisklasse.
- **NEU:** OpcSecurity als generische Basisklasse für Sicherheitseinstellungen von Client und Server Anwendungen.
- **NEU:** OpcClient leitet von OpcApplicationInstance ab.
- **NEU:** OpcServerBase leitet von OpcApplicationInstance ab.
- **NEU:** Zentralisierte Anwendungskonfiguration (innerhalb des Frameworks) von Client / Server Anwendungsinstanzen durch Einsatz der OpcApplicationInstance.
- **NEU:** OpcTransport als generische Klasse zur Konfiguration der Transport-Einstellungen, welche jetzt Teil der IOpcApplicationInstance Schnittstelle ist.
- **NEU:** OpcSecurity als generische Klasse zur Konfiguration der Security-Einstellungen, welche jetzt Teil der IOpcApplicationInstance-Schnittstelle ist.
- **NEU:** Die Eigenschaften Security und Transport wurden der IOpcApplicationInstance Schnittstelle

hinzugefügt.
- **NEU:** Das Ereignis CertificateValidationFailed wurde der IOpcApplicationInstance Schnittstelle hinzugefügt.
- **NEU:** OpcSecurityPolicyCollection implementiert ab sofort einen öffentlichen Konstruktor.
- **VERÄNDERT:** Die Klasse OpcClientTransport wurde entfernt (ersetzt durch die neue OpcTransport Klasse).
- **VERÄNDERT:** Die Klasse OpcServerTransport wurde entfernt (ersetzt durch die neue OpcTransport Klasse).
- **VERÄNDERT:** Die standardmäßig verwendeten Server-Informationen die von der OpcServerBase bereitgestellt werden bevorzugen einen vordefinierten ApplicationName.
- **VERÄNDERT:** OpcSecurityPolicyCollection stellt nicht länger einen Konstruktor bereit, der eine OpcServerBase-Instanz erwartet. Die zugehörige Owner Eigenschaft wurde ebenso entfernt.

2.5.7.0 (freigegeben am 06.02.2019)

- **NEU:** OpcClient.WriteNode akzeptiert jetzt das direkte Schreiben von null als Node-Wert, ohne diesen zuvor in einer OpcValue Instanz zu „verpacken“.
- **NEU:** Die OpcServer Klasse stellt neue Konstruktoren-Überladungen bereit, welche die Standard-Adresse bestehend aus „opc.tcp“ + „localhost“ + „4840“ (der Standardport) verwenden.
- **NEU:** Der gesamte Node-Tree can jetzt „inline“ mittels neuer Konstruktoren auf OpcObjectNode und OpcFolderNode erzeugt werden. Hierzu können die direkten Kindknoten der Knoten direkt beim Erstellen ihrer Elternknoten übergeben werden.
- **NEU:** Ab sofort ist sichergestellt, dass Read/Write Operationen auf Attributen abgebrochen werden, wenn benutzerdefinierte Read/Write Callbacks ein „Bad“-Ergebnis liefern. Im Falle von „Good“-Ergebnissen wird die Read/Write Operationen wie gehabt fortgesetzt.
- **NEU:** API deren Signatur ausschließlich „nodeId:string“ zur Identifizierung einer Node verwendet interpretiert „inline“ den übergebenen String-Wert um eine fließendere Verwendung der Framework API zu gewährleisten.
- **VERÄNDERT:** Anwendungszeritifkate werden nicht länger unter .\CertificateStores\Trusted, sondern (standardmässig) unter .\CertificateStores\App gespeichert, um die Performance beim Start der Anwendung auf der Suche nach dem Anwendungszertifikat zu steigern.
- **VERÄNDERT:** Es wird nicht länger die LinkTime, sondern die LastWriteTime der Assembly bei der Ermittlung der OpcServerInfo einer Assembly verwendet.
- **BEHOBEN:** Problem mit nicht übernommenen Initialwert bei Verwendung des OpcDataVariableNode(parent, name, value)-Konstruktors.
- **BEHOBEN:** Prüfungen des Wertes (die z.B. zu BadTypeMismatch führen) werden jetzt beim Schreiben des Value-Attributs durchgeführt.

2.5.6.0 (freigegeben am 19.10.2018 [Client SDK], 06.02.2019 [Client+Server SDK])

- **NEU:** Erweiterte Linker Unterstützung für Android Projekte beim Linken mittels SDK+User Option.
- **NEU:** Erweiterte Standard-Verzeichnis-Auswahl für Zertifikate. Zu erst wird das lokale Verzeichnis auf Schreibrechte geprüft, dann CommonApplicationData und schließlich ApplicationData.
- **NEU:** Zusätzliche Unterstützung des .NET Frameworks 4.7.1 als Zielframework um nicht länger von 12 weiteren System-Assemblies abhängig zu sein. Diese Abhängigkeiten kommen durch einen Fehler im .NET Framework 4.7.1 bei der Referenzierung von .NET Standard Assemblies zustande. Siehe hierzu: https://github.com/Microsoft/dotnet/blob/master/releases/net471/KnownIssues/514195-Targeting%2

0.NET%20Framework%204.7.1%20copies%20extra%20files%20to%20your%20bin%20directory.md

- **BEHOBEN:** Fehler beim Prüfen, ob ein Zertifikat bereits im Zertifikat-Speicher enthalten ist. Der Speicher lieferte immer den Wert Wahr.
- **BEHOBEN:** Fehler beim Browsen unter Verwendung der Linker Option SDK+User in Android Projekten.

2.5.5.0 (freigegeben am 11.10.2018 [Client SDK], 06.02.2019 [Client+Server SDK])

- **NEU:** API für die exklusive Client Entwicklung aus dem OPC UA Advanced extrahiert.
- **NEU:** CertificateRequested Ereignis auf der OpcClient Klasse zum Abrufen eines alternativen Zertifikats in Fällen, in denen ein existierendes Anwendungsinstanz-Zertifikat nicht gefunden oder nicht länger verwendet werden kann. Über dieses Ereignis kann zusätzlich der Grund, warum ein neues Zertifikat benötigt wird, abgerufen werden. Zusätzlich ermöglicht das Ereignis die Information des Benutzers des Clients.
- **NEU:** Im Fall, dass nicht auf das Verzeichnis „.\CertificateStores\“ (das Standard-Verzeichnis für Zertifikate) schreibend zugegriffen werden kann, wird der Pfad „$(AppData)\OPC Foundation\CertificateStores\“ anstelle des Standard-Verzeichnisses verwendet um die Zertifikate zu verwalten.
- **NEU:** Erhöhte Standard-Laufzeit für Zertifikate: von 12 Monaten (= 1 Jahr) auf 600 Monate (= 50 Jahre).
- **NEU:** Reduzierte geforderte Standard-Schlüssellänge von 2048 auf 1024 um weiterhin ältere Zertifikate zu unterstützen.
- **NEU:** Reduzierte Assembly-Größe und reduzierte Abhängigkeiten durch Reorganisation der intern verwendeten Lizenzierungs-Mechanismen für .NET Standard.
- **NEU:** Im Falle einer fehlgeschlagene Bindung des Sockets wird nicht länger die Exception behandelt noch eine allgemeine erzeugt - stattdessen wird diese an den Aufrufer weitergereicht.
- **NEU:** Binding Eigenschaft auf der OpcServer Klass welche eine Instanz der neuen OpcServerBinding Klasse bereitstellt.
- **NEU:** OpcServerBinding (OpcServerBase.Binding Eigenschaft) welche die Möglichkeit bietet einen Geltungsbereich beim Binden der Adresse anzugeben. Standardmässig wird (wie zuvor) auf jede beliebige IPv4 und auf jede beliebige IPv6 Adresse mit der angegebenen Portnummer gebunden.
- **NEU:** Die Low-Level Read / Write Operationen des Nodemanagers stellen ab sofort OpcNodeAccessTokens mit Operation Result-Informationen bereit.
- **NEU:** Dokumentation für OpcSubscription, OpcMonitoredItem, OpcMonitoredItemEventArgs, OpcMonitoredItemEventHandler und weiterer API bezüglich dem Thema Subscriptions.
- **VERÄNDERT:** Standardwert von MinimumCertificateKeySize von 2048 auf 1024 um kleiner Schlüssellängen älterer Zertifikate zu unterstützen.
- **VERÄNDERT:** Wenn das Anwendungszertifikat einer Client-Instanz mittels Certificate-Eigenschaft geändert wird, dann wird der Store-Path-Type des Zertifikat-Speichers für Anwendungszertifikate auf „System“ geändert. Um weiterhin den Store-Path-Type „Directory“ zu verwenden, muss dieser manuell geändert werden. Zusätzlich muss dann sichergestellt sein, dass das Zertifikat auch im Zertifikat-Speicher für Anwendungszertifikate enthalten ist - nach Änderung des Store-Path-Types auf „Directory“.
- **BEHOBEN:** OpcServer.Security.AutoAcceptUntrustedCertificates wurde bei der Verwendung von Secured Endpoint Policies nicht zur automatischen Annahme eines nicht vertrauenswürdigen Zertifikats berücksichtigt.
- **BEHOBEN:** Problem beim automatischen Ermitteln der Startzeit der Node-Historie, wenn diese Anzahl-basiert gelesen wird.

- **BEHOBEN:** Problem beim Vergleich von bisher nicht-vergleichbaren Monitored Item und Session Instanzen.
- **BEHOBEN:** Problem beim Schreiben von Rohdaten welche in einem OpcValue „verpackt“ transport wurden.
- **BEHOBEN:** Problem bei der Interpretation von AutoAcceptUntrustedCertificates Server Anwendungsinstanzen.
- **BEHOBEN:** Fehler der zum Verwerfen des Store-Paths des Zertifikat-Speichers für Anwendungszertifikate geführt hat, wenn manuell ein Anwendungszertifikat festgelegt wurde.
- **BEHOBEN:** Verwendung des falschen Startwertes beim Lesen von historischen Werten ohne explizite Angabe einer Startzeit.

2.5.4.0 (freigegeben am 27.08.2018)

- **NEU:** Unterstützung der automatischen History-Erzeugung im NodeHistorian durch von Clients geschriebenen Werten (AutoUpdateHistory Eigenschaft).
- **NEU:** OpcNodeHistorian unterscheidet Node-Änderungen und nimmt nur noch Änderungen des Value Attributs in die Historie auf.
- **NEU:** Verbesserte Code Dokumentation der Server-seitigen Subscription / MonitoredItem API.
- **NEU:** Server.OpcSubscription und Server.OpcMonitoredItem Instanzen sind ab sofort vergleichbar.
- **BEHOBEN:** OpcVariableNode.IsHistorizing wird nicht länger nach dem Aufruf von OpcNodeManager.CreateNodes zurückgesetzt.
- **BEHOBEN:** Problem beim Schreiben von Variablen-Knoten mit undefinierten DataType Attribut (funktionierte nicht und führte zum Fehler).
- **VERÄNDERT:** OpcNode.BeforeApplyChanges und OpcNode.AfterApplyChanges werden nur noch ausgelöst, wenn Änderungen bestehen (siehe OpcNode.HasPendingChanges).

2.5.3.0 (freigegeben am 21.08.2018)

- **NEU:** Automatische Wert-Ausrichtung nach dem Lesen eines Node-Wertes (Server-seitig) unter Verwendung des vom Client geforderten Encodings / Werte-Bereiches.
- **NEU:** Neue OpcClientTransport Klasse, Zugriff über OpcClient.Transport. Zur Konfiguration von Einstellungen für Timeouts und Grenzen.
- **NEU:** Neue OpcServerTransport Klasse, Zugriff über OpcServer.Transport. Zur Konfiguration von Einstellungen für Timeouts und Grenzen.
- **BEHOBEN:** Problem bei der Validierung von X509 Identity Tokens aufgrund der Verwendung falscher Zertifikatsdaten.

2.5.2.5 (freigegeben am 07.08.2018)

- **NEU:** NuGet Abhängigkeiten wurden entfernt, die dem Projekt des Benutzers unnötig aufgezwungen wurden. (in .NET Standard)
- **NEU:** Der Inhalt der *.deps.json wurde vereinfacht. (in .NET Standard)

2.5.2.4 (freigegeben am 31.07.2018)

- **BEHOBEN:** Problem beim Senden von globalen Events mittels OpcServer und OpcNodeManager Instanzen.
- **NEU:** Unterstützung von Null-OpcName Instanzen in der OpcNominalNodeIdFactory.

- **NEU:** OpcContext stellt ab sofort immer eine gültige OpcNodeIdFactory Instanz bereit (mindestens die Instanz die mittels OpcNodeId.Factory Eigenschaft festgelegt wurde).

2.5.2.3 (freigegeben am 23.07.2018)

- **NEU:** NuGet aktualisiert.

2.5.2.2 (freigegeben am 23.07.2018)

- **BEHOBEN:** InvalidCastException wenn die Assembly per Assembly.LoadFrom() geladen wurde.

2.5.2.1 (freigegeben am 13.07.2018)

- **BEHOBEN:** DllNotFoundException wenn versucht wurde auf bestimmte Typen zuzugreifen, bevor ein OpcClient oder OpcServer erstellt wurde.

2.5.2.0 (freigegeben am 06.07.2018)

- **NEU:** .NET Standard / .NET Core Lizenzierung und Support wurde freigegeben.

2.5.1.1 (freigegeben am 03.07.2018)

- **NEU:** Demo-Lizenz wurde erneuert, um eine neue Lizenzperiode bereitzustellen.

2.5.1.0 (freigegeben am 15.06.2018)

- **NEU:** MimeType-Unterstützung im OpcFileNode und in OpcFileInfo
- **NEU:** Verbesserte Performance beim Zugriff auf OpcContext-Verweise in OpcNodes (intern).
- **NEU:** Verbesserte Performance und verbesserter Speicherbedarf beim Zugriff auf Kindknoten einer OpcNode (intern).
- **NEU:** Standardmäßig vorbereitete Konfigurations-Knoten für historische Daten werden jetzt on-demand erzeugt.
- **NEU:** Interne Querverweise wurden aufgelöst und schwerfällige Callbacks aufgelöst und durch ein Provider-Pattern ersetzt.
- **NEU:** Die Schnittstelle IOpcNodeInfo wurde eingeführt um das Erzeugen von temporären OpcNode-Instanzen hinfällig zu machen und um die Leistung wie auch den Speicherbedarf zu verbessern.
- **NEU:** Verwendung der Schnittstelle IOpcNodeInfo anstelle der eigentlichen Node immer dann, wenn der Verweis auf einen Node nicht für die weiteren Operationen / Auswertungen notwendig ist.
- **NEU:** Einführung eines Caches für benutzerdefinierte Datentypen, damit ein wiederholtes Reflektieren dessen nicht länger nötig ist. Dies steigert die Leistung.
- **NEU:** Ausnahmen bei der Ausführung von CreateNodes(…) werden nicht länger „gefangen“, sondern an den Aufruf von OpcServer.Start weitergereicht.
- **VERÄNDERT:** IsNodeAccessible(…, IOpcNode node) wurde zu IsNodeAccessible(…, IOpcNodeInfo node)

2.5.0.3 (freigegeben am 24.05.2018)

- **BEHOBEN:** Problem mit fehlenden externen Verweisen wenn mehrere Node Manager den gleichen

benutzerdefinierten Node als Wurzel für ihre Nodes verwenden.

2.5.0.2 (freigegeben am 23.05.2018)

- **BEHOBEN:** Problem mit fehlenden Knoten wenn mehrere Node Manager auf den gleichen Systemknoten verweisen.

2.5.0.1 (freigegeben am 16.05.2018)

- **BEHOBEN:** NullReferenceException in OpcMethodNode wenn ein vordefinierter Knoten als Eltern-Knoten des Methoden-Knotens verwendet wurde.
- **BEHOBEN:** Nicht länger im Adressraum des Servers angezeigte OpcDataVariableNode Knoten, wenn der Konstruktor(parent : IOpcNode, name : OpcName) verwendet wurde.
- **VERÄNDERT:** Beispiel-Projekte wurden auf die C# Sprachversion 6.0 zurückgestuft um sicherzustellen, dass die Bespiele auch in älteren Versionen des Visual Studios kompilierbar sind.

2.5.0.0 (freigegeben am 10.05.2018)

- **NEU:** Unterstützung zur Erstellung von Knoten mittels benutzerdefinierten Knoten Identifizierer.
- **NEU:** Vereinheitlichung der Konstruktoren-Überladungen aller Node-Klassen.
- **NEU:** Viele der bisher fehlenden Code Documentationen der Node-Klassen wurden ergänzt.
- **NEU:** Für jede Eigenschaft eines Knotens wurde die ensprechende XyNode Eigenschaft implementiert, welche die Referenz auf den Eigenschafts-Knoten bereitstellt welcher den Wert der bisher verfügbaren Xy Eigenschaften repräsentiert.
- **NEU:** Unterstützung der späten Initialisierung von Knoten welche es erlaubt Xy Eigenschafts-Werte festzulegen, welche bei der Erzeugung der Eigenschafts-Knoten beim Start des Servers automatisch auf diese angewendet werden.
- **NEU:** Komplette Überarbeitung der Klassen OpcNodeId und OpcNamespace. Teilweise Neuimplementierung beider Klassen zur Sicherstellung der maximalen Kompatibilität zu Foundation definierten Knoten Identifizierer Standards einschließlich der Features die durch das Framework geboten werden.
- **NEU:** OpcNodeId unterstützt Identifizierer mittels Byte-Arrays, Guid's und vorzeichenlosen Integer Werten.
- **NEU:** OpcNodeId drückt die Art des Identifizierers in der String-Repräsentation seines Identifizierer-Wertes über OpcNodeId.ToString aus.
- **NEU:** OpcNodeId unterstützt UTF8 codierte Sonderzeichen in URL formatierten Knoten Identifizierern.
- **NEU:** OpcNodeId unterstützt das Parsen von Foundation formatierten Knoten Identifizierern.
- **NEU:** OpcNodeId bietet weitere Eigenschaften, wie OriginalString und OriginalFormat.
- **NEU:** Überarbeitete OpcName Implementierung um nicht länger eine Kombination von BrowseName, DisplayName und SymbolicName in einer Instanz / Klasse zu vereinen.
- **NEU:** IOpcNode, OpcNode und OpcNodeInfo stellen jetzt direkte Eigenschaften für den DisplayName (und SymbolicName) einer Node bereit.
- **NEU:** OpcNodeInfo unterstützt das Browsing der Kinder und Eltern mittels Kombination von OpcNodeCategory Enumerations-Werten.
- **NEU:** OpcClient.SubscribeEventRaise Methoden und OpcSubscribeEventRaise unterstützen jetzt eine OpcEventFilter Instanz für die Ereignisfilterung.
- **NEU:** Vorhandene API bezüglich UA Alarm + Events wurde überarbeitet und hinsichtlich einer fließenden und intuitiven Schnittstelle einschließlich partieller späten Initialisierung (zur

Performance-Steigerung) angepasst.
- **NEU:** OpcEventRaiseEventArgs wurde überarbeitet und stellt über die Event Eigenschaft Snapshot-Informationen unter Verwendung der Raw-Daten des Events einer EventNode bereit.
- **NEU:** MonitoredItem Eigenschaft wurde zu OpcDataChangeEventArgs und OpcEventRaiseEventArgs hinzugefügt.
- **NEU:** Ereignisse DataChangedReceived und EventRaiseReceived wurden zur OpcSubscription hinzugefügt.
- **NEU:** OpcEventType Enumeration und OpcEventTypes Klasse für den vereinfachten Zugriff auf die verschiedenen Eventtypen.
- **NEU:** Unterstützung für unterschiedliche Zustandsautomaten- und Zustands-/Übergangsvariablen-Knoten.
- **NEU:** DialogRequested Ereignis wurde der OpcClient Klasse hinzugefügt um einen einfachen Weg zur Verarbeitung von OpcDialogConditionNode Instanzen bereitzustellen.
- **NEU:** Unterstützung von Foundation spezifischen Eventtypen: OpcConditionNode, OpcAcknowledgeableConditionNode, OpcDialogConditionNode, OpcAlarmConditionNode, OpcDiscreteAlarmNode, OpcLimitAlarmNode, OpcOffNormalAlarmNode, OpcSystemOffNormalAlarmNode, OpcTripAlarmNode, OpcExclusiveLimitAlarmNode, OpcNonExclusiveLimitAlarmNode, OpcExclusiveDeviationAlarmNode, OpcExclusiveLevelAlarmNode, OpcExclusiveRateOfChangeAlarmNode, OpcNonExclusiveDeviationAlarmNode, OpcNonExclusiveLevelAlarmNode und OpcNonExclusiveRateOfChangeAlarmNode.
- **BEHOBEN:** Bezeichnung der Methode GetOutpuArguments der OpcMethodNodeInfo wurde zu GetOutputArguments geändert.
- **BEHOBEN:** Problem mit DBNull Werten wenn kein Argumentwert bei Methodenknoten definiert wurde.
- **VERÄNDERT:** Falschgeschriebene Member wurde korrigiert.
- **VERÄNDERT:** Vereinheitlichte API durch das Hinzufügen von weiteren selbstbeschreibenden Bestandteilen zu Klassennamen und Delegatnamen. Im Wesentlichen wurde die Bezeichnung von Data auf DataSet, DataItem auf DataSetItem und EventRaise auf Event geändert.
- **VERÄNDERT:** OpcNodeId.ToString formattiert die Informationen jetzt mittels OpcNodeIdFormat.Foundation Format.
- **VERÄNDERT:** Reihenfolge der Subscription-Ereignisse wurde von MonitoredItem → Client → Subscribtion zu MonitoredItem → Subscription → Client geändert. Um eine logischere Abarbeitungsreihenfolgen beim Event-Bubbling sicherzustellen.
- **VERÄNDERT:** OpcMonitoredItem.ReceivedDataChange wurde zu OpcMonitoredItem.DataChangeReceived umbenannt.
- **VERÄNDERT:** OpcMonitoredItem.ReceivedEventRaise wurde zu OpcMonitoredItem.EventRaiseReceived umbenannt.

Kompatibilitätsbrüche (durch Umbenennungen):

- OpcSubscribeEventRaise → OpcSubscribeEvent
- OpcDataChangeEventArgs → OpcDataChangeReceivedEventArgs
- OpcDataChangeEventHandler → OpcDataChangeReceivedEventHandler
- OpcEventRaiseEventArgs → OpcEventReceivedEventArgs
- OpcEventRaiseEventHandler → OpcEventReceivedEventHandler
- OpcNotificationEventArgs → OpcNotificationReceivedEventArgs
- OpcNotificationEventHandler → OpcNotificationReceivedEventHandler
- OpcDataChangeCallback → OpcDataChangeReceivedCallback
- OpcEventRaiseCallback → OpcEventReceivedCallback
- IOpcNotificationData → IOpcNotificationDataSet

- OpcNotificationData → OpcNotificationDataSet
- OpcNotificationDataCollection → OpcNotificationDataSetCollection
- OpcNotificationDataReadOnlyCollection → OpcNotificationDataSetReadOnlyCollection
- OpcNotificationDataItem → OpcNotificationDataSetItem
- OpcNotificationDataItemCollection → OpcNotificationDataSetItemCollection
- OpcNotificationDataItemReadOnlyCollection → OpcNotificationDataSetItemReadOnlyCollection
- OpcDataChange → OpcDataChangeSet
- OpcDataChangeItem → OpcDataChangeSetItem
- OpcEventRaise → OpcEventDataSet
- OpcEventRaiseItem → OpcEventDataSetItem
- OpcMonitoredItem
    - .EventRaiseReceived → EventReceived
    - .LastEventRaise → LastEvent
    - .OnEventRaiseReceived → OnEventReceived
- OpcSubscription
    - .EventRaiseReceived → EventReceived
    - .ReceivedEventRaiseCallback → ReceivedEventCallback
    - .OnEventRaiseReceived → OnEventReceived
- OpcClient
    - .EventRaiseReceived → EventReceived
    - .OnEventRaiseReceived → OnEventReceived
    - .SubscribeEventRaise → SubscribeEvent

2.3.2.0 (freigegeben am 13.03.2018)

- **BEHOBEN:** Problem mit fehlerhaften DebuggerDisplayAttribute in OpcVariableNode und OpcNodeInfo.
- **BEHOBEN:** Problem beim Lesen / Schreiben des DataType-Attributes von OpcVariableNodes.
- **VERÄNDERT:** OpcVariableNode wurde als abstrakte Basisklasse festgelegt.
- **VERÄNDERT:** Mehrfache Foundation Node-Referenzen wurden entfernt um die Speicherbedarf zu senken.
- **NEU:** Erweitertung der Implementierung der OpcVariableNode speziell für Enumerations-Datentypen.
- **NEU:** Neue Nodes OpcTypeNode und OpcDataTypeNode. Die OpcDataTypeNode unterstützt die Definition von benutzerdefinierten Enumerations-Datentypen.
- **NEU:** Erweiterte Unterstützung beim Browsing für Enumerations-Datentypen. Unter Einsatz der neuen OpcTypeNodeInfo.GetEnumMembers() Methode werden alle Informationen über den Auflistungstypen bereitgestellt (siehe hierzu auch die neue OpcTypeNodeInfo.IsEnum Eigenschaft).

2.3.1.1 (freigegeben am 26.02.2018)

- **BEHOBEN:** Problem beim Laden einer OpcApplicationConfiguration aus einer Datei aufgrund des fehlenden DataContract-Attributes auf der OpcApplicationConfiguration Klasse, welches bisher bei älteren Versionen des Frameworks nicht benötigt wurde.
- **NEU:** Weitere Konstruktoren-Überladungen in OpcEngineeringUnitInfo um die Initialisierung für spezifische Einheiten zu vereinfachen.
- **NEU:** ToString Überladungen in OpcEngineeringUnitInfo und OpcValueRange.
- **NEU:** CommonCode Eigenschaft in OpcEngineeringUnitInfo + statische GetCommonCode(int unitId) Methode.

- **NEU:** Das DefaultNamespaceUri der OpcEngineeringUnitInfo wird verwendet, wenn der Konstruktor ohne expliziten Namespace Parameter verwendet wird. Das ist nicht der Fall, wenn der Standard-Konstruktor verwendet wird.

2.3.1.0 (freigegeben am 22.02.2018)

- **ACHTUNG:** Beginnend mit dieser Version muss eine Referenz zum NuGet Paket Portable.BouncyCastle (mindestens V1.8.1.3) oder eine Referenz zur BouncyCastle.Crypto Assembly (mindestens V1.8.1.146) zum Projekt hinzugefügt werden.
- **ACHTUNG:** Wird eine Server Adresse mit dem Schema 'https' verwendet, dann muss eine Referenz zu den NuGet Paketen 'Microsoft.AspNetCore.Server.Kestrel (1.1.3)' und 'Microsoft.AspNetCore.Server.Kestrel.Https (1.1.3)' zum Projekt hinzugefügt werden.
- **NEU:** Einführung des Range Datentypen als OpcValueRange.
- **NEU:** Einführung des EUInformation Datentypen als OpcEngineeringUnitInfo.
- **NEU:** Einführung des neuen Node-Typen AnlogItemType als OpcAnalogItemNode.
- **NEU:** Unterstützung für DataValues welche die Datentypen Range und EUInformation verwenden.
- **NEU:** Erweiterer Browsing-Support beim Browsen von Nodes der AnalogItemType-Definition mittels neuer OpcAnalogItemNodeInfo.
- **NEU:** Upgedated auf die Zielframework Version .NET Framework 4.6.

2.3.0.0 (freigegeben am 13.02.2018)

- **NEU:** Upgedated auf den OPC UA Foundation Stack V1.03.351.0.
- **NEU:** Neue Sicherheits-Option für Client / Server zur Konfiguration von zusätzlichen Zertifikat-Validierungsregeln (siehe OpcClient/OpcServer.Security.CertificateValidationRules).
- **VERÄNDERT:** Das Framework unterstützt nicht länger Windows basierte Authentifizierung mittels WinLogon-API bei der OPC UA Server Authentifizierung.
- **VERÄNDERT:** Anwendungszertifikate werden jetzt mittels SHA256 erstellt (der neue Standardwert der von der Foundation vorgeschrieben wird). Dies beeinflusst keine bisherigen Anwendungen. Diese Änderung hat nur Einfluss darauf, wenn ein neues Zertifikat erstellt wird.

2.2.2.0 (freigegeben am 11.12.2017)

- **NEU:** Erstellung von Filtern wurde mittels Operatoren-Überladung vereinfacht um eine natürlichere API zu erhalten.
- **NEU:** OpcText wurde eingeführt um lokalisierbare und für den Menschen verständliche String-Werte zu repräsentieren.
- **BEHOBEN:** Konfigurierte OperationTimeout wurde im OpcClient.Connect() nicht für das Discovering der Server-Endpunkte verwendet.
- **VERÄNDERT:** OpcValue → ToString Implementierung wurde vereinfacht, sodass nur noch der String „null“ für Null-Referenzen oder Value.ToString() geliefert wird.

2.2.1.0 (freigegeben am 06.09.2017)

- **NEU:** Hinzufügen der Unterstützung von unvollständigen Zertifikatsketten, wenn das Gegenüber ein von einem unbekannten Aussteller unterzeichnetes Zertifikat vorlegt. Das ist beispielsweise beim Versuch einer Verbindung zu einer Siemens SIMOTION der Fall.
- **NEU:** Erweitern der Unterstützung von fehlgeschlagenen Zertifikatsüberprüfungen durch

Verwendung des CertificateValidationFailed Ereignisses in der OpcClient und OpcServer Klasse.

2.2.0.0 (freigegeben am 01.09.2017)

- **NEU:** Hinzufügen des GetReferenceType, um den zugehörigen OpcReferenceType für einen gegebenen OpcNodeId des Typs Reference zu entfernen.
- **VERÄNDERT:** Entfernen der Spezialbehandlung des Attributs DataType im OpcNodeInfo.Attribute(). Das bedeutet, dass nicht länger ein OpcDataType Wert zur Verfügung gestellt wird. Stattdessen wird der datentypabhängige OpcNodeId zur Verfügung gestellt.
- **NEU:** Implementierung von grundlegender Unterstützung von Objektnodes und ausgeweiteter Unterstützung von Variablennodes, ebenso Implementierung von generischer Unterstützung von Typennodes (einschließlich ObjectTypes, VariableTypes und DataTypes) während des Browsens von Nodes.
- **NEU:** Bei der Benutzung von OpcVariableNodeInfo, OpcObjectNodeInfo und OpcTypeNodeInfo ist es nun möglich, den Addressraum mithilfe von Zusatzinformationen über die Nodes zu browsen. Das beinhaltet z.B. auch die Typinformation eines Variablennodes mitsamt all seinen typspezifischen Metadaten einschließlich Über- und Untertypen.
- **NEU:** Bereitmachen des Frameworks für die Benutzung unter .NET Standard und .NET Core 2.0.
- **NEU:** Einführen der LicenseInfo Eigenschaft auf allen Licenser Klassen, um etwas Information über die verwendeten Lizenzbedingungen zur Verfügung zu stellen.

2.1.0.0 (freigegeben am 01.08.2017)

- **NEU:** Upgedated auf den OPC UA Foundation Stack V1.3.342.0.

2.0.1.1 (freigegeben am 05.06.2017)

- **NEU:** RegisterAddress / UnregisterAddress Methoden und neue Addresses Methode, um die verschiedenen Basisadressen einer OpcServerBase Instanz beizubehalten.
- **BEHOBEN:** Falsche Bezeichnung von IsNodeAccessable im OpcNodeManager zu IsNodeAccessible.
- **BEHOBEN:** Problem mit deformierten Nodenamensräumen beim Verwenden eines Namenraum-URI, der nur nach Schema aufgebaut wird, wenn der OpcNamespace.ToString() aufgerufen wird (= FormatException).
- **BEHOBEN:** Problem mit der falschen impliziten Interprätation von Statuscodes als Ergebnisinformationen.

2.0.1.0 (freigegeben am 24.05.2017)

- **NEU:** Hinzufügen von vielen Details in einer Nachricht einer OpcException und in der Beschreibung einer OpcStatus / OpcResult Instanz, um die Quelle / den Grund / das Ergebnis einer Operation besser darzustellen.

2.0.0.0 (freigegeben am 21.05.2017)

- **NEU:** Verändern der the Assembly Version zu 2.0.0.0.
- **NEU:** Implementieren von OpcCertificateSettings, um ein Anwendungsinstanz-Zertifikat zu konfigurieren, das durch den OpcCertificateManager erstellt werden kann.
- **BEHOBEN:** Problem mit falsch konfigurierten Anwendungszertifikaten beim Verwenden eines

Speicherpfads im Falle dessen, dass das Zertifikat unter diesem Pfad nicht gefunden werden kann.

- **BEHOBEN:** Problem mit dem falsch gespeicherten Benutzername-Passwort-Paar in der UserName ACL (Access Control List), das zu einer fehlgeschlagenen Bebutzerauthentifizierung führte.
- **BEHOBEN:** Problem beim Erstellen eines OpcNodeId unter Verwendung des string+int32 Paares, bei dem der int32 nicht für den Namensraumindex des neuen Node Identifiers benutzt wurde.
- **BEHOBEN:** Problem mit OpcServerBase.Certificate und OpcClient.Certificate im Fall, dass das Zertifikat nicht im ApplicationStore des Zertifikatspeichers gespeichert ist.
- **NEU:** Einführen von OpcClientCertificateStores (verfügbar über OpcClient.CertificateStores) und OpcServerCertitificateStores (verfügbar über OpcServerBase.CertificateStores), um einen fließenden API für Client und Server zum Verwalten und Warten ihrer Zertifikatspeicher zur Verfügung zu stellen.
- **NEU:** Einführen von OpcClientServices (verfügbar über OpcClient.Services), um einen dienstbasierten API zum Aufbau der verschiedenen, vom Client genutzten Dienste zur Verfügung zu stellen.
- **ENTFERNT:** „OpcValue.SourceLabel“ und „OpcValue.ServerLabel“. Das beinhaltet auch, dass die „OpcValueLabel“ Klasse nicht länger existiert. Die davon zur Verfügung gestellten Eigenschaften werden nun direkt von der „OpcValue“ Klasse bereitgestellt. Benutzen Sie einfach die „OpcValue.ServerXy“ und „OpcValue.SourceXy“ Eigenschaften.
- **NEU:** IOpcWriteNodesService unterstützt jetzt ein automatisches Feststellen des Wertes, den der Server für den SourceTimestamp eines OpcValue erwartet. Die automatische Bestimmung kann durch das Festlegen einer expliziten OpcTimestampSource durch Verwenden der IOpcWriteNodesService.TimestampSource Eigenschaft deaktiviert werden. Beachten Sie: Auf die IOpcWriteNodesService Instanz kann durch Verwenden der OpcClient.Services.WriteNodes Eigenschaft zugegriffen werden.
- **NEU:** Entfernen des Benutzens der hinfälligen ResolveNodeId Methode und stattdessen Implementieren der neuen Resolve Methode auf der OpcNodeId, um den voll qualifizierten Node durch Verwenden des Namensraumarrays des Servers auf Anfrage zu bestimmen. Das unterstützt nun den Nodezugriff unter Verwendung des voll qualifizierten Node Identifiers ohne das Vorwissen des Indexes des zugehorigen Nodenamensraums.
- **NEU:** Implementieren einer zusätzlichen Überladung der UpdateCertificate Methode, um manuell zu definieren, ob das Zertifikat erstellt werden soll, falls ein Zertifikat fehlt oder ungültig ist. Bisher war dies nur durch die Verwendung der statischen AutoCreateCertificate Eigenschaft möglich.
- **NEU:** Implementieren der neuen Eigenschaften OpcClient.Certificate (um direkt ein benutzerdefiniertes Clientzertifikat zu definieren) und OpcClient.Security (um Client Sicherheitsoptionen innerhalb einer Klasse einzurichten).
- **VERÄNDERT:** Umbenennen der Eigenschaft „OpcSecurity.Policies“ zu „OpcSecurity.EndpointPolicies“, um dem „OpcClientSecurity.EndpointPolicy“ Modell zu entsprechen.
- **VERÄNDERT:** Definieren des Standardwerts von „OpcCertificateManager.AutoCreateCertificate“ als true.
- **NEU:** Implementieren der neuen Eigenschaften in Opc.UaFx.Server.OpcSecurity: AutoAcceptUntrustedCertificates und AutoCreateCertificate.
- **NEU:** Implementieren der neuen OpcServer.Certificate Eigenschaft, um demselben API Modell zu entsprechen, das der OpcClient implementiert.
- **VERÄNDERT:** Verhalten bei der Auswahl eines Endpunkts, auf den sich verbunden werden soll, verändert, damit es nicht länger verhandelbar ist, falls eine explizite Endpunktstrategie definiert ist.
- **VERÄNDERT:** Definieren des Standards von AutoAcceptUntrustedCertificates in der SecurityConfiguration als true.
- **NEU:** Implementieren des Unterstützens eines DefaultNodeManager - dies ist entweder der erste Manager in der Liste der definierten Manager, oder der intern definierte Standardnodemamager, der

den Namensraum als „<serveraddress>/nodes/“ definiert. Nun gibt es auch die Möglichkeit, nur eine Rückmeldemethode zu definieren, um die Nodes für den Adressraum des Standardnodemanagers zur Verfügung zu stellen.

- **VERSCHOBEN:** „OpcClient.PreferredPolicy“ Eigenschaft nach „OpcClient.Security.EndpointPolicy“.
- **VERSCHOBEN:** „OpcClient.UseDomainChecks“ Eigenschaft nach „OpcClient.Security.VerifyServersCertificateDomains“ und Standardwert auf false vgeändert.
- **VERSCHOBEN:** „OpcClient.UserIdentity“ Eigenschaft nach „OpcClient.Security.UserIdentity“ und dessen Typ vom Foundationtyp IUserIdentity zu OpcUserIdentity geändert.
- **VERSCHOBEN:** „OpcClient.UseOnlySecureEndpoints“ Eigenschaft nach „OpcClient.Security.UseOnlySecureEndpoints“ und Standartwert auf false geändert.
- **ENTFERNT:** Veraltete „OpcClient.UseAutoNodeIdResolution“ Eigenschaft.
- **NEU:** Überarbeiten / Überprüfen der OpcIdentity Ableitungen und Definieren der OpcCertificateIdentity, OpcWindowsIdentity, OpcServerIdentity und OpcClientIdentity als neue Ableitungen von OpcUserIdentity (bisher benutzten sie die OpcIdentity nur als ihre Basisklasse). Ebenfalls Verändern ihrer Konstruktionsverhalten, um den Prozess der Identitästerstellung auf Client- und Serverseite zu vereinfachen.
- **BEHOBEN:** OpcRequestType.Call wurde bisher als OpcRequestType.Cancel definiert. Das führte zu unfunktionellem Verwenden des OpcRequestType.Call, weil es wie der OpcRequestType.Cancel verwendet wurde.
- **ENTFERNT:** OpcCertificateManager.AutoCreateCertificate Eigenschaft, weil es nun als Instanz unter Verwendung von OpcClient.Security.AutoCreateCertificate oder OpcServer.Security.AutoCreateCertificate verwendet wird. Diese Eigenschaften sind auch standardmäßig auf den Wert true gesetzt.
- **ENTFERNT:** OpcServer.Security.Owner Eigenschaft, um überflüssige Informationen zu reduzieren und den API zu vereinfachen.
- **UMBENANNT:** „Server.OpcSecurity“ zu „Server.OpcServerSecurity“.
- **UMBENANNT:** „OpcServerSecurity.Anonymous“ zu „OpcServerSecurity.AnonymousAcl“
- **UMBENANNT:** „OpcServerSecurity.Certificate“ zu „OpcServerSecurity.CertificateAcl“
- **UMBENANNT:** „OpcServerSecurity.IssuedToken“ zu „OpcServerSecurity.IssuedTokenAcl“
- **UMBENANNT:** „OpcServerSecurity.UserName“ zu „OpcServerSecurity.UserNameAcl“
- **UMBENANNT:** „OpcClient.PreferredLocales“ zu „OpcClient.Locales“
- **UMBENANNT:** „OpcClient.ReceivedNotification“ zu „OpcClient.NotificationReceived“
- **UMBENANNT:** „OpcClient.OnReceivedNotification“ zu „OpcClient.OnNotificationReceived“
- **UMBENANNT:** „OpcSubscription.ReceivedNotification“ zu „OpcSubscription.NotificationReceived“
- **UMBENANNT:** „OpcSubscription.OnReceivedNotification“ zu „OpcSubscription.OnNotificationReceived“
- **VERSCHOBEN:** OpcNodeCommand von Opc.UaFx.Client nach Opc.UaFx.Services.
- **VERSCHOBEN:** OpcNodeAttributeCommand von Opc.UaFx.Client nach Opc.UaFx.Services.
- **UMBENANNT:** „OpcNodeCommand“ zu „OpcNodeServiceCommand“.
- **UMBENANNT:** „OpcNodeAttributeCommand“ zu „OpcNodeAttributeServiceCommand“.
- **UMBENANNT:** OpcStatusCode Mitglieder mit „Bad“, „Good“ und „Uncertain“ Präfixen wie die OPC Faindation sie verwendet, um Statuscodes in diese drei Kategorien zu unterteilen.
- **NEU:** Implementieren von zusätzlichen ReadNode Überladungen, um den numerischen Node Identifier zusammen mit einem Namensraumindex und zu lesendem Attribut zu definieren.
- **NEU:** Implementieren von neuen ReadNodes Methoden, um mehrere Nodes unter Verwendung eines spezifischen Attributs zu lesen.

1.6.5.2 (freigegeben am 07.02.2017)

- **VERÄNDERT:** Entfernen von impliziten Interprätationen eines numerischen Node Identifiers, der als String verlüsselt ist.

1.6.5.1 (freigegeben am 08.12.2016)

- **NEU:** Implementieren eines erweiterten Timeout Handlings für Read History Anfragen des OpcClients. Das ist notwendig, um dem Server zusätzlich Zeit zu verschaffen, um die historischen Daten zu sammeln, falls dies mehr Zeit als gewöhnlich benötigt und der Server aufeinanderfolgende Weiterführungspunkte bereitstellt, solange bis die Datensammlung abgeschlossen ist.

1.6.5.0 (freigegeben am 07.12.2016)

- **NEU:** Erweitern der Node History Navigation im Falle eines festgelegten Zeitfenstert (in dem StartTime und EndTime nicht gleichgesetzt sind mit MinDate). Daher werden historische Nodewerte auf jede Seitenanfrage eingesammelt, solange der Server einen Weiterführungspunkt bereitstellt bis entweder der Server keinen Weiterführungspunkt mehr bereitstellt oder bis die Seite ihre Seitengröße erreicht hat.

1.6.4.0 (freigegeben am 12.10.2016)

- **NEU:** Updaten der IOpcNode Interfacedefinition hinsichtlich des momentanen öffentlichen OpcNode API.
- **NEU:** Implementieren der neuen Methode Children() : IEnumerable<IOpcNode> in den IOpcNode, um physikalische Kinder im Nodebaum zu bestimmen.

1.6.3.0 (freigegeben am 06.10.2016)

- **NEU:** Implementieren der neuen Methode OpcFileMethods.IsFileNode, um zu bestimmen, ob auf einen Node wie auf einen FileType zugegriffen werden kann.
- **BEHOBEN:** Out of memory Problem bei der Verwendung von OpcFile.ReadAllText und OpcFile.ReadAllLines.
- **NEU:** Verbessern der Leistung von OpcFile.ReadAllText, um den Text nicht länger durch die Zeilen einer Datei zu konstruieren, stattdessen benutzt es jetzt den binären Strom einer Datei.
- **NEU:** Allgemeine Optimierung des Tempos beim Lesen und Schreiben von Dateikomponenten durch Verwenden der clientkonfigurierten MaxByteStringLength.
- **BEHOBEN:** Problem beim browsen von Nodes, die Remotenodes enthalten, was zur Ausnahme 'Cannot cast an absolute ExpandedNodeId to a NodeId. Use ExpandedNodeId.ToNodeId instead.' führte.

1.6.2.0 (freigegeben am 04.10.2016)

- **NEU:** Implementieren der neuen Methode CreateTempCertificate, die auch in Fällen verwendet wird, in denen AutoCreateCertificate auf true gestellt ist und in denen das Zertifikatsgenerator-Dienstprogramm nicht installiert ist. Daher muss das Zertifikasgenerator-Dienstprogramm nicht länger zum Testen / Entwickeln neben OPC Anwendungen implementiert sein.
- **BEHOBEN:** Problem beim Öffnen von Dateien zum Schreiben - die Zugriffsmaske wurde falsch verifiziert.
- **NEU:** Implementieren des neuen Session Resource Manager, um alle sitzungsbezogenen Ressourcen

zu entsorgen, nachdem eine Sitzung abläuft oder schließt. So eine Ressource ist der interne Dateihandle, wenn via OPC auf Dateinodes zugegriffen wird. Falls der Client die Verbindung zum Server aus irgendeinem Grund verwirft, gibt der Session Manager jetzt alle durch die Sitzung gewonnenen Ressourcen wieder frei.

1.6.1.0 (freigegeben am 14.06.2016)

- **NEU:** Implementieren der OpcNodeManager.Browse Methode, um benutzerdefiniertes Browsen in unterklassifizierten Szenarios zu unterstützen.

1.6.0.0 (freigegeben am 25.08.2016)

- **NEU:** Ausgrenzen von Attributen ohne Wert aus dem automatischen Updaten der Ursache des Wertes, wenn ein Nodewert geschrieben wird. Umgehungen, damit ein expliziter OpcValue benutzt werden kann, sind nun hinfällig.
- **NEU:** Einführen der neuen Ableitung OpcBrowseNodeContext, besonders für das Browsen von Nodes, und Reduzieren des bisherigen OpcNodeContext auf eine eher generische Implementierung für weiteres Subklassifizieren. Der neue OpcBrowseNodeContext beinhaltet nun den Node Identifier des Nodes, von dem die Browse-Operation ursprüngich eingeleitet wurde.
- **NEU:** Einführen der OpcValue.As<T> Methode, um den repräsentierten Wert als einen spezifischen Typ abzufragen, indem der Wert in den von T. spezifizierten Typen umgewandelt wird.
- **NEU:** Implementieren der neuen OpcResult Klasse, um ServiceResult Instanzen zu repräsentieren.
- **NEU:** Implementieren der Unterklasse OpcMethodContext von OpcContext, um Systeminformationen zu bestimmen, die sensibel gegenüber Methodenaufrufen sind, einschließlich des aufgerufenen Methodennodes und seines Ziels. Dies beinhaltet auch das Speicher des Ergebnisses des Aufrufs der Rückmeldemethode.
- **NEU:** Implementieren der Unterstützung der Dateisystemzugägnlichkeit für OpcFileNode Objekte aus Clientsicht. Das beinhaltet eine einfache 1:1 API Schicht, implementiert durch die OpcFileMethods Klasse, einen SafeOpcFileHandle, um sicherzustellen, dass zugeteilte Dateihandles freigegeben werden, die OpcFile Klasse zum Implementieren einer OPC-basierten System.IO.File Klasse wie „set of service“ Methoden, die OpcFileInfo Klasse zum Implementieren einer OPC-basierten System.IO.FileInfo Klasse wie „object“ und die OpcFileStream Klasse zum Implementieren eines OPC-basierten System.IO.FileStream wie „object“, der von der System.IO.Stream Klasse abstammt und daher von allen StreamReader und StreamWriter Klassen benutzt werden, die dich das .NET Framework bereitgestellt werden.
- **NEU:** Implementieren des neuen OpcPropertyNode und OpcPropertyNode<T>.
- **VERÄNDERT:** Der Typ OpcVariableValue benötigt nun einen generischen Typparameter für den repräsentierten Wertetyp. Die OpcVariableNode Klasse wurde bezüglich dieses Typparameters aufgenommen.
- **NEU:** Implementieren des neuen OpcFileNode, inklusive seiner child Nodes wie die Methodennodes Nodes Open, Close, GetPosition, SetPosition, Read und Write.

1.5.11.7 (freigegeben am 28.07.2016)

- **BEHOBEN:** Problem mit Remotenode Identifiern im OpcNodeId.
- **BEHOBEN:** Problem mit dem Vergleichen von Node Identifiern verschiedener Typen im OpcNodeId.

1.5.11.6 (freigegeben am 17.06.2016)

- **NEU:** Implementieren der Kindmethode in OpcNodeInfo, um die Nachfolger eines Nodes durch Verwenden seines OpcName auszulesen. Dies beinhaltet den BrowseName, DisplayName und SymbolicName eines Nodes.
- **NEU:** Implementieren der Elternmethode in OpcNodeInfo, um die Vorfahren eines Nodes durch Verwenden seines OpcName auszulesen. Dies beinhaltet den BrowseName, DisplayName und SymbolicName eines Nodes.
- **NEU:** Implementieren der Kindmethode in OpcNodeInfo, um die Nachfolger eines Nodes einer spezifischen OpcNodeCategory auszulesen.
- **NEU:** Implementieren der Elternmethode in OpcNodeInfo, um die Vorfahren eines Nodes einer spezifischen OpcNodeCategory auszulesen.

1.5.11.5 (freigegeben am 13.06.2016)

- **NEU:** Implementieren eines impliziten Cast Operators für ExpandedNodeId's.
- **NEU:** Implementieren einer zusätzlichen Get Methodenüberladung im OpcNamespace, um Nodenamensräume durch die Verwendung von URI oder Index auszulesen.
- **BEHOBEN:** Problem mit ExpandedNodeId's in OpcValue's und deren Handling wurde vereinfacht simplified.

1.5.11.4 (freigegeben am 30.05.2016)

- **NEU:** Minimierung von Lock Bedingungen beim Lesen / Schreiben von Nodes.

1.5.11.3 (freigegeben am 02.05.2016)

- **NEU:** Demo-Lizenz wurde erneuert, um eine neue Lizenzperiode bereitzustellen.

1.5.11.2 (freigegeben am 18.04.2016)

- **BEHOBEN:** Problem mit überwachten Elementen beim lesen des Initialwertes eines Nodes (der Server lief im Falle von ungefilterten überwachten Elementen in eine NullReferenceException).

1.5.11.1 (freigegeben am 31.03.2016)

- **NEU:** Implemented the Changed event on OpcStatus.
- **BEHOBEN:** Issue with the not synchronized OPC Status from SDK to the foundation stack (on OpcValue and OpcVariableNode).

1.5.11.0 (freigegeben am 23.03.2016)

- **NEU:** OpcNamespace, der die gesamte Namensrauminformation eines Node Identifiers repräsentiert (Namespace Index und Namespace Uri).
- **NEU:** IOpcNamespaceResolver, der durch OpcNodeManager, OpcClient und OpcServer implementiert wird. Diese Schnittstelle stellt verspätete gebundene Namensraumauflösung zur Verfügung, nachdem der OpcServer gestartet oder der OpcClient verbunden wurde. Jeder OpcNodeId, dessen Namensraum aufgelöst wurde, stellt die voll qualifizierte URL des Namensraums

zusammen mit dem Node Identifier zur Verfügung.
- **NEU:** OpcNodeId.Namespace Eigenschaft, um den verbundenen OpcNamespace bereitzustellen.
- **VERÄNDERT:** Umbenennen von OpcAttribute.Historizing zu OpcAttribute.IsHistorizing.
- **NEU:** OpcNode: Implementieren einer Nodeveränderungsüberwachung über die Eigenschaften HasPendingChanges und PendingChanges, inklusiver der Methode IsChangePending.
- **NEU:** Implementieren von Historyunterstützung von OpcVariableNode und seine Unterklassen.
- **NEU:** Ausweiten von Konstruktorüberladungen der OpcStatusCollection für ein weitaus einfacheres Erstellen einer neuen Instanz durch OpcStatusCodes.
- **NEU:** HDA Methoden auf dem Client zum Lesen und Updaten der Node History.
- **NEU:** HDA Schnittstelle auf dem Server, um die benötigte Logik zum Handlen des Zugriffs auf historische Daten bereitzustellen.
- **NEU:** HDA Dienste: IOpcReadNodesHistoryService und IOpcUpdateNodesHistoryService.
- **NEU:** HDA befehle: OpcCreateNodeHistory, OpcDeleteNodeHistory, OpcDeleteNodeHistoryAtTime, OpcDeleteNodeHistoryModified, OpcReadNodeHistory, OpcReplaceNodeHistory und OpcUpdateNodeHistory. Alle davon sind Unterklassen von OpcNodeHistoryCommand.

1.5.10.0 (freigegeben am 03.02.2016)

- **ADDED:** OPC Watch Anwendung zum Assembly-Archiv.

1.5.9.1 (freigegeben am 25.01.2016)

- **BEHOBEN:** Problem mit einer falsch geöffneten Sitzung, obwohl sie im Falle einer Neuverbindung wiederverwendet werden kann.

1.5.9.0 (freigegeben am 21.01.2016)

- **BEHOBEN:** Mögliche Multithreading-Probleme in einem Sitzungs-KeepAlive im OpcClient.
- **VERÄNDERT:** Das BreakDetected Ereignis wird nun nur ausgelöst, wenn der Neuverbindungs-Timeout gleich Null ist.
- **BEHOBEN:** Falls eine OPC Client Sitzung geöffnet wird, die schon entsorgt wurde, konnte das in manchen Fällen zu einem unerwünschten Neuverbinden zum Server aufgrund einer verworfenen ObjectDisposedException durch die bereits entsorgte Client Sitzung führen.

1.5.8.0 (freigegeben am 18.01.2016)

- **NEU:** Updaten der Lizenzierungslogik.

1.5.7.1 (freigegeben am 14.01.2016)

- **BEHOBEN:** Problem mit dem kleinstmöglichen PublishingInterval in der OpcSubscription Klasse.

1.5.7.0 (freigegeben am 09.01.2016)

- **BEHOBEN:** Problem mit der Nullreferenzausnahme im OpcCertificateManager.CreateCertificate, wenn ein applicationUri als Nullreferenz (Nothing in Visual Basic)behandelt wird.
- **BEHOBEN:** Problem mit nicht eindeutig generierten OpcNodeId's für InputArguments und OutputArguments der OpcMethodNode Instanzen. Das führte zu dem Verhalten, dass nur die

Argumente des am neuesten bereitgestellten OpcMethodNode beim Browsen / Lesen der Methodenargumenteigenschaften veröffentlicht wurden.
- **VERÄNDERT:** Der OpcNodeId für InputArgument und OutputArgument Nodes von OpcMethodNode Instanzen, für mehr Informationen werfen Sie eien Blick auf das vorherige Thema.
- **VERÄNDERT:** Die OpcReadOnlyNodeCollection Klasse stellt jetzt IOpcNode Instanzen bereit statt der internen NodeState Foundationinstanzen.
- **NEU:** Ein OpcNodeId kann jetzt aus OpcName Instanzen konstruiert werden, deren SymbolicName eine Nullreferenz (Nothing in Visual Basic) ist. In diesem Fall wird statt des SymbolicName der BrowseName verwendet.
- **NEU:** Die OpcReadNodesRequest liefert nun standardmäßig beide Zeitstempel (Quelle und Server) zurück.
- **NEU:** Imlementieren der neuen OpcNodeManager Methoden: AddNode, RemoveNode und IsNodeAccessable.
- **BEHOBEN:** Problem mit zeitweisen ACE (Access Control Entires)-Duplikaten, falls ein ACE angefragt wird, der bereits in eine ACL hinzugefügt wurde.

1.5.6.8 (freigegeben am 17.12.2015)

- **BEHOBEN:** Problem mit verpasstem TimeStamp Update auf dem Schreiben von Nodewerten.
- **BEHOBEN:** Problem mit verpassten TimeStamp Daten auf dem Lesen von Nodewerten.

1.5.6.7 (freigegeben am 17.12.2015)

- **NEU:** Implementieren der ChangePassword Methode in OpcServerIdentity, um das Benutzerpasswort zu ändern, nachdem eine Identität angelegt wurde.
- **NEU:** Implementieren der geschützten SystemContext Eigenschaft im OpcNodeManager, um einen alternativen Kontext bereitzustellen, in dem systeminterne Veränderungen vorgenommen werden können.
- **NEU:** Die OpcWriteNode Klasse setzt nun automatisch den Ursprungszeitstempel beim Schreiben eines Nodes.
- **VERÄNDERT:** Das OpcValueLabel ist nun nicht weiter als Struct definiert, sondern als Klasse.

1.5.6.6 (freigegeben am 16.12.2015)

- **NEU:** Implementieren von AddNode und RemoveNode im OpcNodeManager, um das dynamische Bereitstellen von Nodes nach dem Aufruf von CreateNodes zu unterstützen.

1.5.6.5 (freigegeben am 10.12.2015)

- **NEU:** Hinzufügen von zusätzlichen Sitzungsinformationen zu OpcOperationContext beim Aufruf von IsNodeAccessable im OpcNodeManager.

1.5.6.4 (freigegeben am 09.12.2015)

- **NEU:** Implementieren der IsNodeAccessable Methode im OpcNodeManager, um benutzerdefiniert zu spezifizieren, ob auf einen Node von innerhalb der Ansicht zugegriffen werden kann.

1.5.6.3 (freigegeben am 08.12.2015)

- **BEHOBEN:** Problem mit ACE-Duplicaten beim manuellen Hinzufügen von zusätzlichen Einträgen auf Nachfrage.

1.5.6.2 (freigegeben am 12.10.2015)

- **BEHOBEN:** Problem beim Lesen und Schreiben von OpcValue Instanzen und Arrays as Nodewerte auf Client- und Serverseite.
- **BEHOBEN:** Problem beim anonymen benutzerlogin auf der Serverseite beim Erstellen eines OpcSession Objekts.
- **NEU:** Implementieren einer Umleitung für den Fall, dass der Stringwert, der an den Konstruktor (string, int) der OpcNodeId Klasse weitergeleitet wird, einen numerischen Identifier angibt. Nun wird ein OpcNodeId mit dem numerischen Identifier initialisiert statt mit dem Stringwert.

1.5.6.1 (freigegeben am 08.09.2015)

- **BEHOBEN:** Problem, wenn Anonymous und UserName ACLs gleichzeitig aktiviert sind.
- **NEU:** Implementieren von zusätzlichen CreateCertificate Überläufen im OpcCertificateManager.

1.5.6.0 (freigegeben am 07.09.2015)

- **VERÄNDERT:** Umbenennen der BuiltInType Eigenschaft zu DataType in der OpcAttributeInfo.
- **NEU:** Browsen des DataType Attributs resultiert nun in einem Mitglied des OpcDataType Aufzählungswertes.
- **NEU:** Implementieren des OpcValue als Ersatz für die DataValue Klasse.
- **NEU:** Implementieren der statischen CreateCertificate Methode im OpcCertificateManager, um Zertifikate von der Stange zu erstellen.
- **NEU:** Die OpcStatus Klasse stellt nun die neue OpcStatusCode Aufzähung als Code bereit.
- **NEU:** Implementieren von Unterstützung der OpcDataTypes: ExtensionObject, Value und Variant.
- **VERÄNDERT:** Umbenennen von DataValue zu Value in OpcDataTypes.
- **NEU:** Implementieren von Unterstützung, um das Browsen mithilfe eines Methodennodes direkt zu beginnen.
- **VERÄNDERT:** Verschieben von OpcDiscoveryClient nach Opc.UaFx.Client Namensraum.
- **BEHOBEN:** Problem mit der NullReferenceException, wenn ungecachte Nodes gebrowst werden.
- **NEU:** Implementieren von Unterstützung der Verwendung einer Nullreferenz für eine Anwendungs-URI, um Zertifikate zu erstellen.
- **NEU:** Implementieren einer neuen Abstraktionsschicht für verschiedene OPC Server Typen (OpcServerBase).
- **NEU:** Implementieren einer neuen OPC Server Spezialisation: OPC Discovery Server.
- **BEHOBEN:** Problem im OPC Client beim Zugriff auf die interne Sitzungsinstanz, obwohle diese verworfen wurde.

1.5.5.3 (freigegeben am 25.08.2015)

- **NEU:** Vereinfachen der Benutzung von OpcVariableClass in Lese- / Schreibrückmeldungen durch das Feststellen des Status und des Zeitstempels (falls kein anderer spezifiziert wurde) der Nodeinstanz.

1.5.5.2 (freigegeben am 24.08.2015)

- **BEHOBEN:** Problem, wenn Anonymous und UserName ACLs gleichzeitig aktiviert sind.
- **NEU:** Entfernen des OpcAccessControlMode Mitglieds 'Default' und Ersetzen seiner Verwendung mit Blacklist.

1.5.5.1 (freigegeben am 24.08.2015)

- **BEHOBEN:** Problem mit dem Null DataType in VariableNodes.

1.5.5.0 (freigegeben am 24.08.2015)

- **NEU:** Implementieren verschiedener Rückmeldeeigenschaften zum Lesen / Schreiben von OpcNode Attributen.
- **NEU:** Implementieren geschützter virtueller Methoden zum Lesen / Schreiben von OpcNode Attributen.
- **NEU:** Implementieren spezifischer Rückmeldeeigenschaften zum Lesen / Schreiben von OpcVariableNode Attributen.
- **NEU:** Implementieren geschützter virtueller Methoden zum Lesen / Schreiben von OpcVariableNode Wertattributen.
- **NEU:** Die OpcSession Klasse stellt nun die Eigenschaften UsedIdentity und SuppliedIdentity bereit.
- **NEU:** Implementieren der Update Methode in OpcStatus.
- **NEU:** Der OpcContext stellt nun eine Identity Eigenschaft bereit.
- **NEU:** Implementieren der neuen OpcMonitoredItem Klasse in den Server Namensraum.
- **NEU:** OnMonitoredItemCreated/Deleted/Modified stellt nun andere Klassen als die der Foundation bereit.
- **VERÄNDERT:** Verschieben der Identity und ImpersonationContext Klassen von Opc.UaFx.Server nach Opc.UaFx Namensraum.

1.5.2.0 (freigegeben am 18.06.2015)

- **NEU:** Ausschluss des automatischen Updates des Endpunkts vor dem Verbinden in Fällen, in denen ein bevorzugter Endpunkt im OpcClient definiert ist.
- **NEU:** Implementieren einer neuen Endpunktidentitätssicherheit in OpcSecurity, um verschiedene Endpunkte für einen oder mehrere ACEs zu aktivieren / deaktivieren. Zusätzlich wurden einige existierende Methoden umbenannt.
- **NEU:** Implementieren des StateChanged Ereignisses in OpcClient.

1.5.1.1 (freigegeben am 16.05.2015)

- **NEU:** Implementieren der InvokeService Method in OpcClient, um unter Verwendung eines try-catch-finally Blocks Dienstroutinen hervorzurufen, damit versichert ist, dass die Statuseigenschaft des Clients auch den Verbindungsstatus im Falle einer Ausnahme behandelt, der besagt, dass die Verbindung ein Timeout war, verloren wurde, usw.
- **BEHOBEN:** Problem mit Nullreferenzen als DisplayName und BrowseName in OpcReferenceDescription.

1.5.1.0 (freigegeben am 15.06.2015)

- **NEU:** Implementieren der SaveCertificate Methode im OpcCertificateManager.
- **NEU:** Überarbeiten der kompletten Argumentenbindung, um auch out und ref Parameter im OpcMethodNode zu unterstützen.
- **NEU:** Implementieren der ByRef Type Unterstützung im OpcMethodNode.
- **NEU:** Implementieren des lokalen Server CertificateValidationFailed Event, um benutzerdefinierte fehlgeschlagende Zertifikatvalidierungsaktionen zu unterstützen.
- **NEU:** Implementieren der neuen OpcNodeInfo, abgeleitet von OpcMethodNodeInfo zum Browsen der Methodennodeargumente.
- **NEU:** Implementieren der neuen OpcCertificateValidationFailedEventArgs und OpcCertificateValidationFailedEventHandler.
- **NEU:** Implementieren der DataType Eigenschaft im OpcDataVariableNode.
- **NEU:** Implementieren der InputArguments und OutputArguments Eigenschaften in der OpcMethodNode Klasse. Zusätzlich Ausweitung der Unterstützung von OpcArgument Instanzen mit dem OpcArgumentAttribute in benutzerdefinierten Delegaten.
- **NEU:** Implementieren von Unterstützung for Nullreferenzen in allen impliziten Cast Operatoren, die mit Referenztypen arbeiten.

1.5.0.0 (freigegeben am 10.06.2015)

- **NEU:** Implementieren einer Logik, um eine servererfüllende Konfiguration beim Start zu generierenlogic.
- **NEU:** Komplette Überarbeitung der Endpunktauswahl über den OpcDiscoveryClient und benutzerdefiniert bevorzugt den OpcSecurityPolicy.
- **NEU:** Verbesserung der automatischen Vervollständigung der Konfiguration, abhängig von ihrem Anwendungstyp.
- **NEU:** Einführen der Transportprofilinformationen.
- **NEU:** Verschieben der Konfoguration der UserTokenPolicies zum OpcServer beim Start und Verbesserung der Nachvollziehbarkeit der IsEnabled Eigenschaft.
- **BEHOBEN:** Problem mit leeren Zersifikatspeicherpfaden.
- **NEU:** Komplette Überarbeitung des Nodereferenzmechanismus, um mehr Informationen bereitzustellen, während die Nodereferenzen erstellt werden und auch OPC Interna aus dem Benutzer API zu entfernen.
- **NEU:** Komplette Überarbeitung der bisher existierenden Nideimplementierungen und Verschieben ihrer Basisklassenreferenz in eine interne Referenz, um den Umgang mit dem Framework zu verbessern und angepasster an CLSCompliant zu werden, indem mehr nützliche APIs bereitgestellt werden.
- **NEU:** Implementieren von Unterstützung von Discovery Server Basisadressenkonfiguration.
- **NEU:** Verbesserung der Konfigurationsstandards, auch beim Laden von Konfiguration durch eine der Load Methoden. Zusätzlich Implementieren einer Standartunterstützung von DiscoveryServerConfigurations.
- **NEU:** Weitere Vereinfachung des API.
- **NEU:** Einführen des Ersatzes für StatusCode → OpcStatus.
- **NEU:** Überarbeiten des OpcDiscoveryClient von Factorymethoden zu Konstruktorbedienung, um eine fließende Schittstelle zwischen allen Server- und Clientklassen zur Verfügung zu stellen.

1.4.0.0 (freigegeben am 09.06.2015)

- **NEU:** Zusammenlegen von Opc.ClientFx.Advanced und Opc.ServerFx.Advanced zu Opc.UaFx.Advanced.

1.0.0.0 (freigegeben am 11.01.2015)

- **NEU:** Erstellen von Opc.ClientFx.Advanced und Opc.ServerFx.Advanced.

Inhaltsverzeichnis

Getestet? Du willst es? 1
Historie 2
2.41.1.0 (freigegeben am 18.01.2024) 2
2.41.0.0 (freigegeben am 02.12.2023) 2
2.40.0.0 (freigegeben am 23.10.2023) 2
2.33.0.0 (freigegeben am 11.09.2023) 3
2.32.0.0 (freigegeben am 28.08.2023) 4
2.31.0.0 (freigegeben am 17.05.2023) 4
2.30.0.0 (freigegeben am 07.12.2022) 5
2.29.0.0 (freigegeben am 16.09.2022) 5
2.28.1.0 (freigegeben am 02.09.2022) 6
2.28.0.0 (freigegeben am 19.08.2022) 6
2.27.0.0 (freigegeben am 27.06.2022) 7
2.26.0.1 (freigegeben am 13.04.2022) 7
2.26.0.0 (freigegeben am 13.04.2022) 8
2.25.0.0 (freigegeben am 22.03.2022) 8
2.24.0.0 (freigegeben am 09.03.2022) 9
2.23.0.0 (freigegeben am 02.03.2022) 10
2.22.0.1 (freigegeben am 25.02.2022) 10
2.22.0.0 (freigegeben am 11.02.2022) 10
2.21.0.0 (freigegeben am 01.02.2022) 10
2.20.4.0 (freigegeben am 14.01.2022) 11
2.20.3.0 (freigegeben am 04.01.2022) 11
2.20.1.0 (freigegeben am 08.10.2021) 11
2.20.2.0 (freigegeben am 20.10.2021) 11
2.20.1.0 (freigegeben am 08.10.2021) 11
2.20.0.0 (freigegeben am 17.09.2021) 12
2.19.2.0 (freigegeben am 10.09.2021) 12
2.19.1.0 (freigegeben am 01.09.2021) 13
2.19.0.0 (freigegeben am 27.08.2021) 14
2.18.5.0 (freigegeben am 13.08.2021) 14
2.18.4.0 (freigegeben am 09.08.2021) 15
2.18.3.0 (freigegeben am 28.06.2021) 15
2.18.2.0 (freigegeben am 17.06.2021) 15
2.18.1.0 (freigegeben am 02.06.2021) 16
2.18.0.0 (freigegeben am 26.05.2021) 16
2.17.0.0 (freigegeben am 04.05.2021) 17
2.16.1.0 (freigegeben am 21.04.2021) 18
2.16.0.0 (freigegeben am 20.04.2021) 18
2.15.0.0 (freigegeben am 31.03.2021) 18
2.14.0.0 (freigegeben am 04.03.2021) 19
2.13.0.0 (freigegeben am 01.03.2021) 19
2.12.3.0 (freigegeben am 17.02.2021) 20
2.12.2.0 (freigegeben am 15.02.2021) 20
2.12.1.0 (freigegeben am 11.02.2021) 20
2.12.0.0 (freigegeben am 04.02.2021) 21
2.11.5.0 (freigegeben am 21.12.2020) 22
2.11.4.0 (freigegeben am 15.12.2020) 23
2.11.3.1 (freigegeben am 27.11.2020) 23
2.11.3.0 (freigegeben am 23.11.2020) 23
2.11.2.0 (freigegeben am 10.11.2020) 24

2.11.1.0 (freigegeben am 05.11.2020) 24
2.11.0.0 (freigegeben am 06.10.2020) 25
2.10.0.3 (freigegeben am 11.09.2020) 26
2.10.0.2 (freigegeben am 09.09.2020) 26
2.10.0.1 (freigegeben am 15.07.2020) 26
2.10.0.0 (freigegeben am 14.07.2020) 26
2.9.2.1 (freigegeben am 08.05.2020) 27
2.9.2.0 (freigegeben am 06.05.2020) 27
2.9.1.0 (freigegeben am 22.04.2020) 28
2.9.0.0 (freigegeben am 01.04.2020) 28
2.8.3.1 (freigegeben am 24.01.2020) 31
2.8.3.0 (freigegeben am 16.01.2020) 31
2.8.2.1 (freigegeben am 13.12.2019) 32
2.8.2.0 (freigegeben am 06.11.2019) 32
2.8.1.3 (freigegeben am 24.10.2019) 32
2.8.1.2 (freigegeben am 23.10.2019) 32
2.8.1.1 (freigegeben am 11.10.2019) 33
2.8.1.0 (freigegeben am 25.09.2019) 33
2.8.0.0 (freigegeben am 18.09.2019) 33
2.7.5.1 (freigegeben am 15.08.2019) 35
2.7.5.0 (freigegeben am 13.08.2019) 35
2.7.4.0 (freigegeben am 07.06.2019) 36
2.7.3.1 (freigegeben am 23.05.2019) 36
2.7.3.0 (freigegeben am 17.05.2019) 36
2.7.2.0 (freigegeben am 10.05.2019) 37
2.7.1.0 (freigegeben am 25.03.2019) 37
2.7.0.2 (freigegeben am 15.03.2019) 38
2.7.0.1 (freigegeben am 15.03.2019) 38
2.7.0.0 (freigegeben am 14.03.2019) 38
2.6.0.0 (freigegeben am 20.02.2019) 39
2.5.7.0 (freigegeben am 06.02.2019) 40
2.5.6.0 (freigegeben am 19.10.2018 [Client SDK], 06.02.2019 [Client+Server SDK]) 40
2.5.5.0 (freigegeben am 11.10.2018 [Client SDK], 06.02.2019 [Client+Server SDK]) 41
2.5.4.0 (freigegeben am 27.08.2018) 42
2.5.3.0 (freigegeben am 21.08.2018) 42
2.5.2.5 (freigegeben am 07.08.2018) 42
2.5.2.4 (freigegeben am 31.07.2018) 42
2.5.2.3 (freigegeben am 23.07.2018) 43
2.5.2.2 (freigegeben am 23.07.2018) 43
2.5.2.1 (freigegeben am 13.07.2018) 43
2.5.2.0 (freigegeben am 06.07.2018) 43
2.5.1.1 (freigegeben am 03.07.2018) 43
2.5.1.0 (freigegeben am 15.06.2018) 43
2.5.0.3 (freigegeben am 24.05.2018) 43
2.5.0.2 (freigegeben am 23.05.2018) 44
2.5.0.1 (freigegeben am 16.05.2018) 44
2.5.0.0 (freigegeben am 10.05.2018) 44
2.3.2.0 (freigegeben am 13.03.2018) 46
2.3.1.1 (freigegeben am 26.02.2018) 46
2.3.1.0 (freigegeben am 22.02.2018) 47
2.3.0.0 (freigegeben am 13.02.2018) 47
2.2.2.0 (freigegeben am 11.12.2017) 47
2.2.1.0 (freigegeben am 06.09.2017) 47

2.2.0.0 (freigegeben am 01.09.2017) 48
2.1.0.0 (freigegeben am 01.08.2017) 48
2.0.1.1 (freigegeben am 05.06.2017) 48
2.0.1.0 (freigegeben am 24.05.2017) 48
2.0.0.0 (freigegeben am 21.05.2017) 48
1.6.5.2 (freigegeben am 07.02.2017) 50
1.6.5.1 (freigegeben am 08.12.2016) 51
1.6.5.0 (freigegeben am 07.12.2016) 51
1.6.4.0 (freigegeben am 12.10.2016) 51
1.6.3.0 (freigegeben am 06.10.2016) 51
1.6.2.0 (freigegeben am 04.10.2016) 51
1.6.1.0 (freigegeben am 14.06.2016) 52
1.6.0.0 (freigegeben am 25.08.2016) 52
1.5.11.7 (freigegeben am 28.07.2016) 52
1.5.11.6 (freigegeben am 17.06.2016) 53
1.5.11.5 (freigegeben am 13.06.2016) 53
1.5.11.4 (freigegeben am 30.05.2016) 53
1.5.11.3 (freigegeben am 02.05.2016) 53
1.5.11.2 (freigegeben am 18.04.2016) 53
1.5.11.1 (freigegeben am 31.03.2016) 53
1.5.11.0 (freigegeben am 23.03.2016) 53
1.5.10.0 (freigegeben am 03.02.2016) 54
1.5.9.1 (freigegeben am 25.01.2016) 54
1.5.9.0 (freigegeben am 21.01.2016) 54
1.5.8.0 (freigegeben am 18.01.2016) 54
1.5.7.1 (freigegeben am 14.01.2016) 54
1.5.7.0 (freigegeben am 09.01.2016) 54
1.5.6.8 (freigegeben am 17.12.2015) 55
1.5.6.7 (freigegeben am 17.12.2015) 55
1.5.6.6 (freigegeben am 16.12.2015) 55
1.5.6.5 (freigegeben am 10.12.2015) 55
1.5.6.4 (freigegeben am 09.12.2015) 55
1.5.6.3 (freigegeben am 08.12.2015) 56
1.5.6.2 (freigegeben am 12.10.2015) 56
1.5.6.1 (freigegeben am 08.09.2015) 56
1.5.6.0 (freigegeben am 07.09.2015) 56
1.5.5.3 (freigegeben am 25.08.2015) 56
1.5.5.2 (freigegeben am 24.08.2015) 57
1.5.5.1 (freigegeben am 24.08.2015) 57
1.5.5.0 (freigegeben am 24.08.2015) 57
1.5.2.0 (freigegeben am 18.06.2015) 57
1.5.1.1 (freigegeben am 16.05.2015) 57
1.5.1.0 (freigegeben am 15.06.2015) 58
1.5.0.0 (freigegeben am 10.06.2015) 58
1.4.0.0 (freigegeben am 09.06.2015) 59
1.0.0.0 (freigegeben am 11.01.2015) 59