KNX Association, XML Scheme Documentation

Document

Association Name, WG	KNX ASSOCIATION
Author(s):	KNX & DEV
Maturity/ Status:	Valid
Version:	1.7
Date:	01.06.2016
Document file name:	KNX-XML Project-Schema-v17 - Description.docx
Number of pages:	42

Acronyms

DEV	Development Companies
KNX	KNX Association
MT5	KNX Manufacturer Tool 5

Referenced documents

[XSD]	XML schema (KNX-Project-Schema-v13.xsd. part of KNX MT5 → Version 5.5)
[DS]	XML DSIG documentation (xmldsig-core-schema.xsd)

List of Changes

Version	Date (DD-MM-YYYY)	Maturity Draft/Valid	Author (Name/Company)	Description
1.0	03.11.2010	WD	A. Hänel; KNXA	Initial public version, derived from KNX internal version 0.43
1.1	02.03.2011	WD	A. Hänel; KNXA	Updates due to export container description and references
1.2	24.06.2011	WD	A. Hänel; KNXA	Updates due to the KNX decision on ex/ import limitations
1.3	10.08.2011	V	A. Hänel; KNXA	Updates due to legal usage of KNX data, no functional changes
1.4	23.05.2012	V	A. Hänel; KNXA	Updates due to change to XML scheme 1.1 (ETS 4.1/ ETS4.2)
1.5	17.10.2014	V	A. Hänel; KNXA	Updates due to change to XML scheme 1.2 (ETS5)
1.6	27.11.2014	V	A. Hänel; KNXA	Updates due to change to XML scheme 1.2 (ETS5)
1.7	01.06.2016	V	A. Hänel; KNXA	Updates due to change to XML scheme 1.3 (ETS5) → Version 5.5

Disclaimer

The document is subject to change without notice.

KNX Association SHALL IN ANY CASE NOT BE LIABLE FOR DIRECT AND INDIRECT DAMAGES ARISING FROM incorrect or missing descriptions in this document, especially when basing software and or hardware developments on the content of this document.

Contents

1 Overview 4
1.1 Document Purpose 4
1.2 Extended Import Restrictions 4
1.3 Extended Import Checks 4
1.4 Validity 4
1.5 Namespaces 4
2 XSD Schema File & KNX Master Data File 5
3 Elements, Types and Attributes 6
3.1 General 6
3.1.1 Element KNX 6
3.1.2 Enumerations 6
3.1.3 Other simpleTypes 12
3.2 Project Data 15
3.2.1 element KNX/Project 15
3.2.2 complexType Project_t 16
3.2.3 General 16
3.2.4 Topology 20
3.2.5 Device Data 22
3.2.6 Building Structure 29
3.2.7 Group Addresses 32
3.2.8 SplitInfos 34
4 IDs and relations 35
4.1 ID naming schema 35
4.1.1 MasterData 35
4.1.2 Manufacturer Data 35
4.1.3 Project Data 36
4.2 Reference Summary 37
4.2.1 Manufacturer Data → Manufacturer Data 37
4.2.2 Project Data → Master Data 37
4.2.3 Project Data → Manufacturer Data 37
4.2.4 Project Data → Project Data 37
5 Transfer files 38
5.1 File extensions 38
5.2 Content 38
5.2.1 Non-XML files 38
5.2.2 Distribution to partial XML files 38
5.2.3 Naming convention 39
5.2.4 Password protection 39

5.3 ETS Container Structure 39
5.3.1 ETS Product Structure 39
5.3.2 ETS Project Structure 41
5.3.3 Password protected projects 42

1 Overview

With introduction of ETS4, the ETS4 and ETS5 ex/- import format for KNX projects and products changed to a standard XML based format (by ETS4/5 exported projects have the file extension *.knxproj).

1.1 Document Purpose

This document describes all necessary elements, types and attributes of the KNX XML Schema [XSD] for an **ETS5** created project. All other –for the project scope not relevant - elements/ attributes might be missing or simply only listed (but not described).

The main use case is to read in (import) ETS5 projects into external tools (e.g. visualizations), but another use case might be to create an ETS5 project from scratch and later import into ETS5 (import is however restricted).

The document **does not** describe how manufacturers create and define products (parameter and/or communication object dependencies and their visibility in correlation with download image creation) to compile valid device configurations outside ETS5. The KNX MT5 exclusively handles this task.

1.2 Extended Import Restrictions

ETS will import projects only from a trusted source, which means

A) the project originates (exported) from ETS itself

B) the project originates from a **KNX member** (and only products of this member are contained in the project)

This is done via a dedicated project signature, in case B the KNX manufacturer shall obtain a unique signature. This implies that an “unreliable” project import - from a source not trusted by ETS - is not possible!

Extended import restrictions implemented in the ETS 4.1, ETS 4.2 and ETS 5.0/ ETS 5.5.

1.3 Extended Import Checks

The ETS5 check on import if a project is valid as regards conformance to the XML conformity (syntax check), i.e. the ETS5 checks if the project format is correct. ETS5 does not check if the saved data inside the file (normally a project/ installation) is a valid project/ installation configuration (semantic check), e.g. if such a project is semantically valid[1].

Hence, it is expected that saved projects & configurations are valid as regards ETS project and installation data integrity.

1.4 Validity

This XML documentation refers to XML scheme V 1.3 (as currently implemented in ETS 5.5).

1.5 Namespaces

The “targetNamespace” is defined as “http://knx.org/xml/project/13”; the prefix **knx** is used here. The schema references the name spaces http://www.w3.org/2001/XMLSchema (prefix **xs**).

[1] This validity covers things such as *KNX project settings used and processed by ETS* up to any *manufacturer device configuration* (with its communication object/ parameter dependencies and visibilities).

2 XSD Schema File & KNX Master Data File

The *KNX XML scheme* is normally defined and described in a file with file extension *.xsd. This file is not part of an ETS5 installation, but of MT5 (the MT5 purpose is to build/ compile valid KNX products and therefore it uses the XML scheme as a basis).

The *KNX master data* contains data definitions, which describe basic KNX system properties as data point types, manufacturer IDs and other things. This data is mandatory for any KNX project and product description. The file normally has the file extension *.xml, the current name is *knx_master.xml*.

For valid owners of the MT (KNX members) it is allowed to use and distribute the *KNX XML scheme* and the *KNX master data* file as part of their own tool chain without any legal restrictions. When this *KNX XML scheme* or the *KNX master data* is updated, it lies within the responsibility of the tool owner to keep his own tool chain up to date.

The information on any update of *KNX XML scheme* will be provided by KNX a few months prior to the official availability of the scheme.

The *KNX master data* will be updated in ETS on demand (online update capability), the corresponding version can be seen in the ETS overview screen.

3 Elements, Types and Attributes

3.1 General

3.1.1 Element KNX

Description	Root element of the XML document.

Children

Name	Description
MasterData	No scope of project part, therefore not detailed here.
ManufacturerData	No scope of project part, therefore not detailed here.
Project	Any number of projects.

Attributes

Name	Type	Use	Default	Description
CreatedBy	xs:string	optional		The tool that created this XML file may include its name here. ETS will write "ETS5".
ToolVersion	xs:string	optional		The tool that created this XML file may include its version here. ETS will write "5.0.xxxx.zzzzz" (xxxx is the build number, zzzzz is the changeset).

3.1.2 Enumerations

3.1.2.1 simpleType Access_t

Type	restriction of **xs:string**
Description	This enumeration encodes the rights for the ETS user to view and modify parameters.
Facets	enumeration None enumeration Read enumeration ReadWrite

3.1.2.2 simpleType GroupAddressStyle_t

Type	restriction of **xs:string**
Description	This enumeration contains the different types of representations of group addresses in ETS4. 2-level and 3-level style are also available in ETS3, the free group address structure is new to ETS4.
Facets	enumeration TwoLevel enumeration ThreeLevel enumeration Free

3.1.2.3 simpleType BuildingPartType_t

Type	restriction of **xs:string**
Description	This enumeration contains the different types of build parts available in the ETS4.

Facets	enumeration Building enumeration BuildingPart enumeration Floor enumeration Stairway enumeration Room enumeration Corridor enumeration DistributionBoard

3.1.2.4 simpleType ComObjectPriority_t

Type	restriction of **xs:string**
Description	This enumeration lists the possible transmission priorities available in the KNX protocol.
Facets	enumeration Low enumeration High enumeration Alert

3.1.2.5 simpleType ComObjectSize_t

Type	restriction of **xs:string**
Description	This enumeration lists the possible data sizes for KNX group communication.
Facets	enumeration 1 Bit enumeration 2 Bit enumeration 3 Bit enumeration 4 Bit enumeration 5 Bit enumeration 6 Bit enumeration 7 Bit enumeration 1 Byte enumeration 2 Bytes enumeration 3 Bytes enumeration 4 Bytes enumeration 5 Bytes enumeration 6 Bytes enumeration 7 Bytes enumeration 8 Bytes enumeration 9 Bytes

	enumeration 10 Bytes enumeration 11 Bytes enumeration 12 Bytes enumeration 14 Bytes enumeration LegacyVarData

3.1.2.6 simpleType CompletionStatus_t

Type	restriction of **xs:string**
Description	Several elements contain a completion status attrubute which might have one of the following values:
Facets	enumeration Undefined enumeration Editing enumeration FinishedDesign enumeration FinishedCommissioning enumeration Tested enumeration Accepted enumeration Locked

3.1.2.7 simpleType Enable_t

Type	restriction of **xs:string**
Description	This enumeration is used for the group object communication flags.:
Facets	enumeration Enabled enumeration Disabled

3.1.2.8 simpleType PropType_t

Type	restriction of **xs:string**
Description	List of interface object property types
Facets	enumeration PDT_CONTROL enumeration PDT_CHAR enumeration PDT_UNSIGNED_CHAR enumeration PDT_INT enumeration PDT_UNSIGNED_INT enumeration PDT_KNX_FLOAT enumeration PDT_DATE

enumeration PDT_TIME
enumeration PDT_LONG
enumeration PDT_UNSIGNED_LONG
enumeration PDT_FLOAT
enumeration PDT_DOUBLE
enumeration PDT_CHAR_BLOCK
enumeration PDT_POLL_GROUP_SETTINGS
enumeration PDT_SHORT_CHAR_BLOCK
enumeration PDT_DATE_TIME
enumeration PDT_VARIABLE_LENGTH
enumeration PDT_GENERIC_01
enumeration PDT_GENERIC_02
enumeration PDT_GENERIC_03
enumeration PDT_GENERIC_04
enumeration PDT_GENERIC_05
enumeration PDT_GENERIC_06
enumeration PDT_GENERIC_07
enumeration PDT_GENERIC_08
enumeration PDT_GENERIC_09
enumeration PDT_GENERIC_10
enumeration PDT_GENERIC_11
enumeration PDT_GENERIC_12
enumeration PDT_GENERIC_13
enumeration PDT_GENERIC_14
enumeration PDT_GENERIC_15
enumeration PDT_GENERIC_16
enumeration PDT_GENERIC_17
enumeration PDT_GENERIC_18
enumeration PDT_GENERIC_19
enumeration PDT_GENERIC_20
enumeration PDT_UTF-8
enumeration PDT_VERSION
enumeration PDT_ALARM_INFO
enumeration PDT_BINARY_INFORMATION
enumeration PDT_BITSET8
enumeration PDT_BITSET16
enumeration PDT_ENUM8

	enumeration PDT_SCALING enumeration PDT_NE_VL enumeration PDT_NE_FL enumeration PDT_FUNCTION

3.1.2.9 simpleType ProjectTracingLevel_t

Type	restriction of **xs:string**
Description	ProjectTracingLevel enumeration
Facets	enumeration None enumeration OperationUsed enumeration Detailed

3.1.2.10 simpleType ToDoStatus_t

Type	restriction of **xs:string**
Description	ToDo status enumeration
Facets	enumeration Open enumeration Accomplished

3.1.2.11 simpleType ApplicationProgramIPConfig_t

Type	restriction of **xs:string**
Description	IPConfig enumeration for the application program
Facets	enumeration Custom enumeration Tool

3.1.2.12 simpleType IPConfigAssign_t

Type	restriction of **xs:string**
Description	Enumeration describing whether IP configuration is performed automatically or by fixed configuration
Facets	enumeration Fixed enumeration Auto

3.1.2.13 simpleType TextEncoding_t

Type	restriction of **xs:string**
Description	This enum may only contain valid codepages!
Facets	enumeration us-ascii enumeration iso-8859-1 enumeration iso-8859-2 enumeration iso-8859-3 enumeration iso-8859-4 enumeration iso-8859-5 enumeration iso-8859-6 enumeration iso-8859-7 enumeration iso-8859-8 enumeration iso-8859-9 enumeration iso-8859-10 enumeration iso-8859-13 enumeration iso-8859-15 enumeration utf-8

3.1.2.14 simpleType RFDeviceMode_t

Type	restriction of **xs:string**
Description	This enum represents the different modes a RF device can have
Facets	enumeration Ready enumeration Multi

3.1.2.15 simpleType SecurityMode_t

Type	Restriction of **xs:string**
Description	This enum represents the different options for secure communication
Facets	enumeration Auto enumeration On enumeration Off

3.1.3 Other simpleTypes

3.1.3.1 simpleType IDREF

Type	**xs:NCName**
Description	This type is used for references to xs:ID. In constrast to the standard XML IDREF type, the referenced element need not be in the same XML file.

3.1.3.2 simpleType IDREFS

Type	**xs:list of knx:IDREF**
Description	This type is used for multiple references to xs:ID, separated by space. In constrast to the standard XML IDREFS type, the referenced elements need not be in the same XML file.

3.1.3.3 simpleType Capabilities_t

Type	**xs:list of knx:Capability_t**
Description	Used to list the actions, an EtsDataHandler is capable of.

3.1.3.4 simpleType String20_t

Type	**xs:string**
Description	Same as xs:string, but restricted to 20 unicode characters.

3.1.3.5 simpleType String50_t

Type	**xs:string**
Description	Same as xs:string, but restricted to 50 unicode characters.

3.1.3.6 simpleType String255_t

Type	**xs:string**
Description	Same as xs:string, but restricted to 255 unicode characters.

3.1.3.7 simpleType LanguageDependentString_t

Type	**xs:string**
Description	This type is used for texts in master or product data that may be translated to different languages.

3.1.3.8 simpleType LanguageDependentString20_t

Type	**xs:LanguageDependentString_t**
Description	Same as LanguageDependentString_t, but restricted to 20 unicode characters.

3.1.3.9 simpleType LanguageDependentString50_t

Type	**xs:LanguageDependentString_t**
Description	Same as LanguageDependentString_t, but restricted to 50 unicode characters.

3.1.3.10 simpleType LanguageDependentString255_t

Type	**xs:LanguageDependentString_t**
Description	Same as LanguageDependentString_t, but restricted to 255 unicode characters.

3.1.3.11 simpleType Regex_t

Type	**xs:string**
Description	Same as string, but must obey the rules of a .NET Regex.

3.1.3.12 simpleType AccessLevel_t

Type	restriction of **xs:unsignedByte**
Description	This type is for specifying the segment access level in LdCtrlDeclarePropDesc.
Facets	minInclusive 0 maxInclusive 15

3.1.3.13 simpleType FloatFormat_t

Type	restriction of **xs:string**
Description	This type is for specifying the DisplayFormat of a Parameter of Type TypeFloat
Facets	[#,]*[0,]+(\.0*)?([eE][+-]?0+)?[#,]*[0,]+(\.0*)?([eE][+-]?0+)?

3.1.3.14 simpleType BitOffset_t

Type	restriction of **xs:unsignedByte**

Description	This type is for specifying the bit offset of parameters. The bit offset is the distance of the most significant bit of the parameter from the most significant bit of the first octet in memory.
Facets	minInclusive 0 maxInclusive 7

3.1.3.15 simpleType Condition_t

Type	**xs:string**
Description	This type is for specifying conditions in When_t. The following values are possible (*number* is an integer value written in decimal notation, **()?+*** are the usual EBNF symbols, □ denotes the space character): A single number — *number* — The condition evaluates to true, if the value of the controlling parameter is numerically equal to the given number. Space-separated list of numbers — *number* **(** □ *number* **)*** — The condition evaluates to true, if the value of the controlling parameter is numerically equal to any one of the given numbers. Comparison expressions — *op number* — Compares the value of the controlling parameter to the given number using one of the comparison operators: = != > < >= <= (note that < > have to be written as < / > in XML attributes) The controlling parameter must be of type TypeNumber or TypeRestriction. In the latter case, the Value attribute is used in the comparison. The planned MT may accept (on import only) also names instead of numbers if the parameter is of type TypeRestriction. But at latest when the data is submitted for registration, these have to be replaced by numeric values since otherwise the registration signature will get invalid on an XML → DB → XML round trip.

3.1.3.16 simpleType Value_t

Type	**xs:string**
Description	This type is for storing parameter values. The different parameter types are encoded as follows: TypeNone — Always the empty string. TypeText — The text value, suitably escaped by character references (e.g. for the tab character) or entity references (e.g. < instead of <). Note that all whitespace characters (newline, tab etc.) must be written as character references, otherwise input normalization would replace them by space characters. TypeNumber — The numeric value, formatted as decimal string. TypeFloat — The numeric value, formatted in scientific notation, with 16 significant digits and 3 exponent digits (regular expression: "-?\d\.\d{15}E[+-]\d{3}"). This corresponds to the conversion value.ToString("E15", CultureInfo.InvariantCulture) in C#. Note: if a Value_t attribute would ever be registration-relevant, care must be taken to ensure that this attribute is reproduced exactly on all data transformations, e.g. when importing the XML into an ETS 4 database and exporting it again. TypeRestriction — The Value attribute of the selected Enumeration option. TypeTime — Same as TypeNumber TypeDate — yyyy-mm-dd TypeIPAddress — IPv4 addresses: decimal dotted notation IPv6 addresses: eight groups of four hexadecimal digits, separated by colons, e.g. 2001:0db8:85a3:0000:0000:8a2e:0370:7334

3.1.3.17 simpleType Guid_t

Type	restriction of **xs:string**
Description	This type is for specifying GUIDs, e.g. the CLSIDs of Plugins.
Facets	pattern \{[0-9A-F]{8}-[0-9A-F]{4}-[0-9A-F]{4}-[0-9A-F]{4}-[0-9A-F]{12}\}

3.1.3.18 simpleType Ipv4Address_t

Type	restriction of **xs:string**
Description	This type is for specifying IP v4 addresses, e.g. the IP routing multicast address.
Facets	pattern ((25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])\.){3}(25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])

3.1.3.19 simpleType RegistrationNumber_t

Type	restriction of **xs:string**
Description	This type is for specifying registration numbers in the format yyyy/n
Facets	pattern \d{4}/\d+

3.1.3.20 simpleType HardwareVersionNumber_t

Type	restriction of **xs:unsignedShort**
Description	This type is for specifying the VersionNumber of a hardware. Restricted to ensure compatibility with ETS3
Facets	minInclusive 0 maxInclusive32767

3.1.3.21 simpleType Aes128Key_t

Type	**xs:string**
Description	Same as xs:string, but restricted to 40 characters. Used to represent a base64-encoded string of an AES128 key.

3.2 Project Data

3.2.1 element KNX/Project

Description	Contains a project.

Type	**knx:Project_t**

3.2.2 complexType Project_t

Description	Contains a project.

Children

Name	Description
ProjectInformation	Contains general information about the project.
Installations	Contains the list of installations within the project.. Most project will just have one Installation. Count of installations must be in [1…16].
AddinData	Contains project related data for Addins
UserFiles	Contains the user files that are appended to the project

Attributes

Name	Type	Use	Default	Description
Id	xs:ID	required		Unique ID of the project in the knxproj container. On export or conversion, this will be constructed as **P-*nnnn***, where: *nnnn* Random 16Bit Identifier, formatted as 4 hexadecimal digits . Must be unique in the knxproj container.

3.2.2.1 element Project_t/UserFiles

Description	Contains the Userfiles
Type	**knx:Userfiles_t**

3.2.2.2 complexType UserFile_t

Description	An element of the Userfile

Attributes

Name	Type	Use	Default	Description
Filename	knx:string255_t	required		The name of the user file
Comment	xs:string	optional		A comment for the user file

3.2.3 General

3.2.3.1 element Project_t/ProjectInformation

Description	Contains general information about the project.

Children

Name	Description
HistoryEntries	Contains history entries entered by the user.
ToDoItems	Contains project related ToDo notes
ProjectTraces	Contains the ProjectTraces
DeviceCertificates	Contains the DeviceCertificates

Attributes	Name	Type	Use	Default	Description
	Name	knx:String50_t	required		Project Name
	GroupAddressStyle	knx:GroupAddressStyle_t	required		Representation of group addresses in this project
	ProjectNumber	knx:String50_t	optional		Optional project number
	ContractNumber	knx:String50_t	optional		Optional contract number
	LastModified	xs:dateTime	optional		Date and time of last modification (UTC)
	ProjectStart	xs:dateTime	optional		Date of project start (UTC)
	ProjectEnd	xs:dateTime	optional		Date of schedules project end (UTC)
	ProjectId	xs:unsignedShort	optional		KNXnet/IP project ID [0 … 4095]. Not used for other media. See KNX standard, Volume 3, Part 8, Chapter 2.
	ProjectPassword	knx:String20_t	optional		Project password. Note that the password is not encrypted in the XML file as password protected projects are stored in encrypted zip containers (see chapter 5.2.4 Password protection).
	Comment	xs:string	optional		Optional comment
	CompletionStatus	knx:CompletionStatus_t	optional	Undefined	Completion status
	ProjectTracingLevel	knx:ProjectTracingLevel_t	optional	None	The Level for ProjectTraces
	ProjectTracingPassword	knx:String20_t	optional		The password for ProjectTracing. This is stored as the first 20 characters of the Base64 encoded string of the salted hash of the original password. “PT-“ is used as salt.
	Hide16BitGroupsFromLegacyPlugins	xs:boolean	optional	false	If true, the project will not use 16 bit groups. This will prevent problems with older plugins that only support 15 bit groups.
	CodePage	knx:TextEncoding_t	optional		Optional CodePage for correct encoding of project related texts.
	BusAccessLegacyMode	xs:Boolean	optional	false	Determines the mode of the buss access
	Guid	xs:string	required		The project guid, used to secure the project data
	LastUsedPuid	xs:int	required		The highest puid that is so far used in the project

3.2.3.2 element Project_t/ProjectInformation/HistoryEntries

Description	List of history entries entered by the user
Children	Name Description **HistoryEntry**

3.2.3.3 element Project_t/ProjectInformation/HistoryEntries/HistoryEntry

Description	History entries entered by the user

Attributes	Name	Type	Use	Default	Description
	Date	xs:dateTime	required		Date and time of the history entry (UTC)
	User	knx:String50_t	optional		User name (optional)
	Text	xs:string	required		Text of the history entry
	Detail	xs:string	optional		Detailed text for the entry

3.2.3.4 element Project_t/ProjectInformation/ProjectTraces

Description	Contains the ProjectTraces
Type	**knx:ProjectTraces_t**

3.2.3.5 complexType ProjectTrace_t

Description	An element of the ProjectTrace

Attributes

Name	Type	Use	Default	Description
Date	xs:datetime	required		The date and time of the trace's creation
UserName	xs:string	required		The name of the user
Comment	xs:string	required		The text of the trace

3.2.3.6 element Project_t/ProjectInformation/DeviceCertificates

Description	Contains the DeviceCertificates
Type	**knx:DeviceCertificates_t**

3.2.3.7 complexType DeviceCertificate_t

Description	An element of the DeviceCertificate

Attributes

Name	Type	Use	Default	Description
SerialNumber	xs:base64Binary	required		The serial number of the device
FDSK	knx:Aes128Key_t	required		The factory default setup key of the device

3.2.3.8 element Project_t/ProjectInformation/ToDoItems

Description	Contains the ToDoItems
Type	**knx:ToDoItems_t**

3.2.3.9 complexType ToDoItem_t

Description	An element of the ToDoItem

Attributes

Name	Type	Use	Default	Description
Description	xs:string	required		The description of the item
ObjectPath	xs:string	optional		The path to the object
Status	knx:ToDoStatus_t	required		The status of the ToDoItem, either "Open" or "Accomplished"

3.2.3.10 element Project_t/AddinData

Description	List of AddinData

3.2.3.11 complexType AddinData_t

Description	An element of the AddinData

Attributes

Name	Type	Use	Default	Description
Name	knx:String50_t	required		The name of the Addin
AddinId	xs:ID	required		The identifier of the Addin

3.2.3.12 complexType BusAccess_t

Description	The information for the bus access

Attributes

Name	Type	Use	Default	Description
Name	xs:string	required		The name of the access
Edi	knx:Guid_t	required		The Guid of the access type
Parameter	xs:string	required		The parameters necessary for the connection

3.2.3.13 element Project_t/Installations

Description	Contains the list of installations within the project.

Children

Name	Description
Installation	Up to 16 installations

3.2.3.14 element Project_t/Installations/Installation

Description	Contains data for one installation

Children

Name	Description
Topology	Contains the topology structure and device data
Buildings	Contains the building structure
GroupAddresses	Contains the group address structure
Trades	Contains the trades structure
BusAccess	Contains the bus access information for the installation
SplitInfos	Contains the split infos for the installation

Attributes

Name	Type	Use	Default	Description
Name	knx:String50_t	required		Name of the installation. If the project contains just one installation, this can be set to an empty string
InstallationId	xs:unsignedShort	optional		KNXnet/IP installation ID [0…15]; not used for other media.

					See KNX standard, Volume 3, Part 8, Chapter 2
	BCUKey	xs:unsignedLong	optional	4294967295	The key used to lock devices supporting authentication.
	IPRoutingMulticastAddress	knx:Ipv4Address_t	optional	224.0.23.12	The multicast address for IP communcation.
	MulticastTTL	xs:byte	optional	16	The time to live for multicast telegrams, i.e.the number of routers the telegram may pass before deletion.
	IPRoutingBackboneKey	knx:Aes128Key_t	optional		For symmetric encryption the AES algorithm with a key length of 128 bit is used. For every IP multicast group, a single encryption key is used. This key is stored in every device of the IP multicast group and has an unlimited lifetime.
	IPRoutingLatencyTolerance	xs:unsingedShort	optional		To prevent replay attacks, the devices shall only accept IP telegrams that were received within a specified time after the telegram was sent. This tolerance can be specified by the user. The latency tolerance is specified in milliseconds.
	IPSyncLatencyFraction	xs:float	optional	0.1	To define the latency for secure IP communication.For futher information, please see KSG 616
	IPRoutingBackboneSecurity	knx:IPRoutingBackboneSecurity_t	optional	Auto	Specifies if the communication via IP is secure or not. Can be either Auto, On or Off. On means the IP communication is performed securely, Off means the IP communication is performed normally. Auto means: If every IP device in the installation has an ApplicationProgram with IsSecureEnabled == true, the communication is performed securely.
	DefaultLine	xs:string	optional		The RefId of the default line.
	CompletionStatus	knx:CompletionStatus_t	optional	Undefined	Completion status
	SplitType	xs:string	optional		Completion status

3.2.4 Topology

3.2.4.1 element Project_t/Installations/Installation/Topology

Description	Contains the topology structure and device data

3.2.4.2 complexType Topology_t

Description	Contains the topology structure and device data
Children	Name Description **Area** Up to 16 Areas **UnassignedDevices** List of devices not assigned to a line

3.2.4.3 element Topology_t/Area

Description	Description of a KNX area				
Children	Name Description **Line** Up to 16 lines				
Attributes	Name	Type	Use	Default	Description
	Id	xs:ID	optional		Unique ID. On export or conversion, this will be constructed as *parid*_**A-***number*, where: *parid* ID of the parent Project and InstallationID sepearted with '-'

				number Unique number of the area within the project. This does not reflect the area address! For converted projects, this corresponds to Area.UniqueNumber in the database schema.
	Name	knx:String255_t	required	Name of the area
	Address	xs:int	required	Area address [0…15]
	Comment	xs:string	optional	User comment
	CompletionStatus	knx:CompletionStatus_t	optional	Completion status
	Description	xs:string	optional	Description of the area
	Puid	xs:int	required	The project wide unique identifier. After deletion of the element, no other element will receive the same Puid.

3.2.4.4 element Topology_t/Area/Line

Description	Description of a KNX line				
Children	Name	Description			
	DeviceInstance	List of devices assigned to the line.			
	AdditionalGroupAddresses	List of additional group addresses that should be included in the filter table of this line's line coupler.			
	BusAccess	Contains the bus access information for the line			
Attributes	Name	Type	Use	Default	Description
	Id	xs:ID	required		Unique ID. On export or conversion, this will be constructed as *parid*_**L-***number*, where: *parid* ID of the parent Project and InstallationID sepearted with '-' *number* Unique number of the line within the project. This does not reflect the line address! For converted projects, this corresponds to Line.UniqueNumber in the database schema.
	Name	knx:String255_t	required		Name of the line
	Address	xs:int	required		Line address [0…15]
	MediumTypeRefId	knx:IDREF	required		Medium type of the line, a reference to MediumType.
	Comment	xs:string	optional		User comment
	DomainAddress	xs:unsignedLong	optional		For open media (PL, RF), the domain address
	CompletionStatus	knx:CompletionStatus_t	optional		Completion status
	Description	xs:string	optional		Description of the line
	Puid	xs:string	required		The project wide unique identifier. After deletion of the element, no other element will receive the same Puid.

3.2.4.5 element Topology_t/Area/Line/DeviceInstance

Description	Represents a device in the project.
Type	**knx:DeviceInstance_t**

3.2.4.6 element Topology_t/Area/Line/AdditionalGroupAddresses

Description	List of additional group addresses that should be included in the filter table of this line's line coupler.
Children	Name Description **GroupAddress** GroupAddress that is not necessarily contained in the project

3.2.4.7 element Topology_t/Area/Line/AdditionalGroupAddresses/GroupAddress

Description	
Attributes	Name Type Use Default Description Address xs:unsignedShort required The address of the GroupAddress

3.2.4.8 element Topology_t/UnassignedDevices

Description	List of devices not assigned to a line
Children	Name Description **DeviceInstance** List of devices assigned to no line.

3.2.4.9 element Topology_t/UnassignedDevices/DeviceInstance

Description	Represents a device in the project.
Type	**knx:DeviceInstance_t**

3.2.5 Device Data

3.2.5.1 complexType DeviceInstance_t

Description	Represents a device in the project.
Children	Name Description **ParameterInstanceRefs** List of parameter instances with non-default values **ComObjectInstanceRefs** List of group communication object instances **ChannelInstances** List of channel instances. ChannelInstances are used **AdditionalAddresses** Additional individual addresses of the device **BinaryData** For use by plugins **IPConfig** The IP configuration of the device **Security** The security configuration of the device **BusInterface** The bus interface of the device

Attributes	Name	Type	Use	Default	Description
	Id	xs:ID	required		Unique ID. On export or conversion, this will be constructed as *parid*_**DI**-*number*, where: *parid* ID of the parent Project and InstallationID sepearted with '-' *number* Unique number of the area within the project. This does not reflect the device address! For converted projects, this corresponds to DeviceInstance.UniqueNumber in the database schema.
	Name	knx:String255_t	optional		Device name
	ProductRefId	knx:IDREF	required		Reference to a Product; must be a child of the Hardware2Progrem element
	Hardware2ProgramRefId	knx:IDREF	optional		Reference to a Hardware2Program
	Address	xs:int	optional		Device address [0…255]
	Comment	xs:string	optional		Device comment
	LastModified	xs:dateTime	optional		Date/time of last modification (UTC)
	LastDownload	xs:dateTime	optional		Date/time of last download (UTC)
	LastUsedAPDULength	xs:unsignedShort	optional		
	ReadMaxAPDULength	xs:unsignedShort	optional		
	ReadMaxRoutingAPDULength	xs:unsignedShort	optional		
	InstallationHints	xs:string	optional		Installation hints, may be plain text or RTF text
	CompletionStatus	knx:CompletionStatus_t	optional	Undefined	Completion status
	IndividualAddressLoaded	xs:boolean	optional	false	true if the IA has been programmed
	ApplicationProgramLoaded	xs:boolean	optional	false	true if the application program has been programmed
	ParametersLoaded	xs:boolean	optional	false	true if the parameters has been programmed
	CommunicationPartLoaded	xs:boolean	optional	false	true if the group communication part has been programmed
	MediumConfigLoaded	xs:boolean	optional	false	true if the PL medium configuration has been programmed
	LoadedImage	xs:base64Binary	optional		The image loaded into the device the last time (used with differential download)
	CheckSums	xs:base64Binary	optional		Check sums read from the device the last time (used with differential download)
	Description	xs:string	optional		Device description.
	IsCommunicationObjectVisibilityCalculated	xs:boolean	optional		If the IsCommunicationObjectVisibilityCalculated flag exists at the DeviceInstance and is "true", the activity for the ComObjectInstanceRefs of this DeviceInstance is already determined for the current DeviceInstance configuration. In this case, the IsActive flag exists at the active ComObjectInstanceRefs elements in the Xml.
	Broken	xs:boolean	optional	false	true if the OnImport handler failed. A broken application program cannot be used in the ETS4.
	SerialNumber	xs:base64Binary	optional		The SerialNumber is used for DownloadIndividualAddressBySerialNumber. This serial number must be provided base64 encoded.
	UniqueId	knx:Guid_t	optional		The unique identifier for the device instance. This is set, if an AddIn requests the identifier and the device instance has none set so far. Otherwise, this unique identifier remains null..
	IsRFRetransmitter	xs:boolean	optional		True if the device instance shall act as a RF retransmitter
	Puid	xs:string	required		The project wide unique identifier. After deletion of the element, no other element will receive the same Puid.
	SequenceNumber	xs:unsignedLong	optional		The value of the last received sender counter. The SequenceNumber is updated during secure online communication.

3.2.5.2 complexType IPConfig_t

Description	IP configuration for the DeviceInstance

Attributes

Name	Type	Use	Default	Description
Assign	knx:IPConfigAssign_t	optional	Auto	If the value is 'Auto', the IP configuration is fetched from DHCP, if the value is 'Fixed', the IP configuration is performed manually
IPAddress	knx:Ipv4Address_t	optional		The IP address of the IP device
SubnetMask	knx:Ipv4Address_t	optional		The subnet mask of the IP device
DefaultGateway	knx:Ipv4Address_t	optional		The default gateway of the IP device
MACAddress	knx:String50_t	optional		The MAC address of theIP device

3.2.5.3 complexType Security_t

Description	Configuration for security elements

Attributes

Name	Type	Use	Default	Description
LoadedIPRoutingBackboneKey	knx:Aes128Key_t	optional		After the download of a device, the encryption key of the IP multicast group is written to the device. The user cannot set the key manually. This encryption key is used for the symmetric encryption within the IP multicast group.
DeviceAuthenticationCode	knx:Aes128Key_t	optional		The device authentication code is generated when the device is instanciated . This is a generated random number (128 bit) and cannot be changed by the user.
LoadedDeviceAuthenticationCode	knx:Aes128Key_t	optional		The device authentication code that was used with the last device downloaded.
DeviceManagementPassword	knx:String20_t	optional		The management password is generated when the device is instanciated. The initial password has a length of 8 elements and consists of lower and upper case letters, numbers and the special characters "+", "-", ",", ".", "#" and "*". The device management password can be changed by the user anytime.
LoadedDeviceManagementPassword	knx:String20_t	optional		The device management password that was used with the last device download.
ToolKey	knx:Aes128Key_t	optional		The tool key for the device.
LoadedToolKey	knx:Aes128Key_t	optional		The tool key used with the last device download.
SequenceNumber	xs:unsignedLong	optional		The value of the last received sender counter. The SequenceNumber is updated during secure online communication.
SequenceNumberTimestamp	xs:dateTime	optional		The timestamp of the last sequence number. This could be used to check how trustworthy a sequence number is.

3.2.5.4 element DeviceInstance_t/BusInterfaces

Description	Contains bus interfaces for the device

Children

Name	Description
BusInterface	The bus interface (can be 1…n)

3.2.5.5 complexType BusInterface_t

Description	Bus interface of the device, only used for devices that have one or more tunnelling server. For more information, please see KSG 616.

Children

Name	Description
Connectors	If the tunnelling server is used for a visualisation, the addresses that shall be visualized can be added here, so that the filter tables are calculated correctly. .

Attributes	Name	Type	Use	Default	Description
	Name	xs:string	optional		The name of the additional address used as a bus interface.
	Description	xs:string	optional		The description for the additional address used as a bus interaface.
	Comment	xs:string	optional		The comment for the additional address used as a bus interface.
	Password	knx:String20_t	optional		The optional password for the tunnelling server..

3.2.5.6 element BusInterface_t/Connectors

Description	Group addresses assigned to the bus interface. Needed for correct calculation of filter tables.

Children	Name	Description
	Connector	Connector to a group address that shall be represented in the calculated filter table.

3.2.5.7 element BusInterface_t/Connectors/Connector

Description	Group addresses assigned to a ComObjectInstanceRef for sending (and receiving)

Attributes	Name	Type	Use	Default	Description
	GroupAddressRefId	knx:IDREF	required		Reference to a GroupAddress

3.2.5.8 element DeviceInstance_t/ParameterInstanceRefs

Description	List of parameter instances with non-default values. If a parameter has its default value, it needs not be listed here.

Children	Name	Description
	ParameterInstanceRef	

3.2.5.9 element DeviceInstance_t/ParameterInstanceRefs/ParameterInstanceRef

Description	Parameter instance

Attributes	Name	Type	Use	Default	Description
	Id	xs:ID	optional		Might be set and used by Plugins. It is recommended to use one of the following methods for constructing the attribute value: • a GUID (without enclosing braces) • *deviceid_paramrefid* where *deviceid* is the Id of the parent Device and *paramrefid* is the Id of the referenced ParameterRef
	RefId	knx:IDREF	required		Reference to a ParameterRef.
	Value	knx:Value_t	optional		The current value

3.2.5.10 element DeviceInstance_t/ComObjectInstanceRefs

Description	List of group communication object instances.

	If a communication object instance has all default settings and no associations, it needs not be listed here.
Children	Name Description **ComObjectInstanceRef**

3.2.5.11 element DeviceInstance_t/ComObjectInstanceRefs/ComObjectInstanceRef

Description	Goup communication object instance
Type	**knx:ComObjectInstanceRef_t**

3.2.5.12 complexType ComObjectInstanceRef_t

Description	Goup communication object instance
Children	Name Description **Connectors**Assigned group addresses

Attributes

Name	Type	Use	Default	Description
Id	xs:ID	optional		The identifier
RefId	knx:IDREF	required		Reference to a ComObjectRef
Text	knx:String255_t	optional		Visible communication object name. If missing, the attribute of the underlying ComObjectRef or ComObject is used
FunctionText	knx:String255_t	optional		Visible communication object function name. If missing, the attribute of the underlying ComObjectRef or ComObject is used
Priority	knx:ComObjectPriority_t	optional		Transmission priority. If missing, the attribute of the underlying ComObjectRef or ComObject is used.
ReadFlag	knx:Enable_t	optional		Read flag. If missing, the attribute of the underlying ComObjectRef or ComObject is used.
WriteFlag	knx:Enable_t	optional		Write flag. If missing, the attribute of the underlying ComObjectRef or ComObject is used.
CommunicationFlag	knx:Enable_t	optional		Communication flag. If missing, the attribute of the underlying ComObjectRef or ComObject is used.
TransmitFlag	knx:Enable_t	optional		Transmit flag. If missing, the attribute of the underlying ComObjectRef or ComObject is used.
UpdateFlag	knx:Enable_t	optional		Update flag. If missing, the attribute of the underlying ComObjectRef or ComObject is used.
ReadOnInitFlag	knx:Enable_t	optional		ReadOnInit flag. If missing, the attribute of the underlying ComObjectRef or ComObject is used.
DatapointType	knx:IDREFS	optional		May be a reference to (one or more) DatapointType or DatapointSubtype. If missing, the attribute of the underlying ComObjectRef or ComObject is used.
Description	xs:string	optional		Description
IsActive	xs:boolean	optional		The IsActive flag is valid if the IsCommunicationObjectVisibilityCalculated flag exists at the DeviceInstance to which this ComObjectInstanceRef belongs and is set to “true”. The IsActive flag of all ComObjectInstanceRefs for this DeviceInstance is then set appropriately and updates may only occur when a value of a ParameterInstanceReference changes.
ChannelId	knx:IDREF	optional		The reference to the ApplicationProgramChannel in which the ComObjectInstance is located. If the ComObjectInstance is located in the ChannelIndependentBlock, the ChannelId is null.

3.2.5.13 element DeviceInstance_t/ChannelInstances

Description	List of channel instances, can be 0…n. ChannelInstances are only available, if PreEts4Style of the referenced ApplicationProgram is false and the ApplicationProgram does not only contain the ChannelIndependentBlock.
Children	Name: **ChannelInstance** — Description: List of channel instances.

3.2.5.14 element DeviceInstance_t/ChannelInstances/ChannelInstance

Description	The channel instances are used to visualize the logical structure of the ComObjectInstances of the device.

Attributes

Name	Type	Use	Default	Description
Id	xs:ID	required		The unique identifier for the ChannelInstance. Is a combination of Device ID and unique Channel ID.
RefId	knx:IDREF	optional		Reference to a Channel in the dynamic part of the ApplicationProgram. If the channel is user defined, the RefId is null.
Name	knx:String255_t	optional		The name of the channel.Can only be edited, if RefId == null (i.e. only names of user defined ChannelInstances can be edited)
Description	knx:String255_t	optional		The description of the channel.
IsActive	xs:boolean	optional		The indicator whether the channel is currently active

3.2.5.15 element ComObjectInstanceRef_t/Connectors

Description	Group addresses assigned to a ComObjectInstanceRef

Children

Name	Description
Send	Sending group address
Receive	Any number of receiving group addresses

3.2.5.16 element ComObjectInstanceRef_t/Connectors/Send

Description	Group addresses assigned to a ComObjectInstanceRef for sending (and receiving)

Attributes

Name	Type	Use	Default	Description
GroupAddressRefId	knx:IDREF	required		Reference to a GroupAddress
Acknowledge	xs:boolean	optional	false	If true, an L2-Ack is produced on PL.

3.2.5.17 element ComObjectInstanceRef_t/Connectors/Receive

Description	Group addresses assigned to a ComObjectInstanceRef for receiving

Attributes

Name	Type	Use	Default	Description
GroupAddressRefId	knx:IDREF	required		Reference to a GroupAddress
Acknowledge	xs:boolean	optional	false	If true, an L2-Ack is produced on PL.

3.2.5.18 element DeviceInstance_t/AdditionalAddresses

Description	Contains additional device addresses used by the device (maximum 254)
Children	Name Description **Address** Device address

3.2.5.19 element DeviceInstance_t/AdditionalAddresses/Address

Description	Additional device address (individual address) used by the device

Attributes

Name	Type	Use	Default	Description
Address	xs:unsignedByte	required		The additional device address (individual address) used by the device.
Name	knx:String255_t	optional		The name of the additional address.
Description	xs:string	optional		The description of the additional address.
Comment	xs:string	optional		A comment for the additional address.

3.2.5.20 element DeviceInstance_t/BinaryData

Description	For use by plugins
Children	Name Description **BinaryData**

3.2.5.21 element DeviceInstance_t/BinaryData/BinaryData

Description	For use by plugins
Children	Name Description **Data** Any data (optional)

Attributes

Name	Type	Use	Default	Description
Id	xs:string	optional		Might be set and used by Plugins. It is recommended to use one of the following methods for constructing the attribute value: • a GUID (without enclosing braces) • *deviceid_id* where *deviceid* is the Id of the parent Device and *id* is the Id of the referenced BinaryData or the suitably escaped name .
RefId	knx:IDREF	optional		Reference to a BinaryData.
Name	knx:String50_t	optional		

3.2.5.22 element DeviceInstance_t/BinaryData/BinaryData/Data

Description	

Type	**xs:base64Binary**

3.2.6 Building Structure

3.2.6.1 element Project_t/Installations/Installation/Buildings

Description	Contains the building structure
Type	**knx:Buildings_t**
Children	Name Description **BuildingPart**

3.2.6.2 complexType Buildings_t

Description	Contains the building structure
Children	Name Description **BuildingPart** Any number of buildings

3.2.6.3 element Buildings_t/BuildingPart

Description	A building. BuildingPart elements directly below Buildings_t will nromally have Type "Campus" or "Building"
Type	**knx:BuildingPart_t**

3.2.6.4 complexType BuildingPart_t

Description	An element of the building structure
Children	Name Description **BuildingPart** Child building parts **DeviceInstanceRef** List of devices in this building part. **Function** List of functions in this building part.

Attributes	Name	Type	Use	Default	Description
	Id	xs:ID	required		Unique ID. On export or conversion, this will be constructed as *parid*_**BP-***number*, where: *parid* ID of the parent Project and InstallationID sepearted with '-' *number* Unique number of the building part within the project.
	Name	knx:String255_t	required		Name
	Type	knx:BuildingPartType_t	required		One of: "Campus", "Building", "BuildingPart", "Floor", "Room", "RoomPart", "DistributionBoard"

	Number	knx:String255_t	optional		Optional number
	Comment	xs:string	optional		Cptional comment
	CompletionStatus	knx:CompletionStatus_t	optional	Undefined	Completion status
	DefaultLine	xs:string	optional		The RefId of the line, to which devices will be added if added to the BuildingPart
	Description	xs:string	optional		Description
	Puid	xs:string	required		The project wide unique identifier. After deletion of the element, no other element will receive the same Puid.

3.2.6.5 element BuildingPart_t/BuildingPart

Description	Child building part.
Type	**knx:BuildingPart_t**

3.2.6.6 element BuildingPart_t/DeviceInstanceRef

Description	References a device contained in a building part.
Type	**knx:DeviceInstanceRef_t**

3.2.6.7 element BuildingPart_t/Function

Description	References a function contained in a building part.
Type	**knx:Function_t**

3.2.6.8 complexType DeviceInstanceRef_t

Description					
Attributes	Name	Type	Use	Default	Description
	RefId	knx:IDREF	required		Reference to DeviceInstance

3.2.6.9 complexType Function_t

Description	A function containing group addresses				
Children	Name	Description			
	GroupAddressRef	List of functions in this building part.			
Attributes	Name	Type	Use	Default	Description
	Id	xs:ID	required		
	Name	knx:String255_t	required		Name
	Type	knx:String255_t	optional		The optional type of the function

Number	knx:String255_t	optional		Optional number
Comment	xs:string	optional		Cptional comment
Description	xs:string	optional		Description
CompletionStatus	knx:CompletionStatus_t	optional	Undefined	Completion status
DefaulGroupRange	xs:IDREF	optional		The RefId of the default GroupRange
Puid	xs:string	required		The project wide unique identifier. After deletion of the element, no other element will receive the same Puid.

3.2.6.10 complexType GroupAddressRef_t

Description: A type containing information of the referenced group address

Attributes:

Name	Type	Use	Default	Description
Id	xs:ID	required		Unique identifier of the GroupAddressRef
RefId	knx:IDREF	required		Reference to GroupAddress
Name	knx:String255_t	required		Name
Role	knx:String255_t	optional		The optional name of the role of that group address
Puid	xs:string	required		The project wide unique identifier. After deletion of the element, no other element will receive the same Puid.

3.2.6.11 complexType Trades_t

Description: Contains the trades structure

Children:

Name	Description
Trade	Any number of trades

3.2.6.12 element Trades_t/Trade

Description: A Trade.

Type: **knx:Trade_t**

3.2.6.13 complexType Trade_t

Description: An element of the trades structure

Children:

Name	Description
Trade	Child Trades
DeviceInstanceRef	List of devices in this trade.

Attributes:

Name	Type	Use	Default	Description
Id	xs:ID	optional		Unique ID.

					On export or conversion, this will be constructed as *parid*_**T-***number*, where: *parid* ID of the parent Project and InstallationID sepearted with '-' *number* Unique number of the Trade within the project.
	Name	knx:String255_t	required		Name of the trade
	Number	knx:String255_t	optional		Optional number
	Comment	xs:string	optional		Cptional comment
	CompletionStatus	knx:CompletionStatus_t	optional	Undefined	Completion status
	Description	xs:string	optional		Description of the trade
	Puid	xs:string	required		The project wide unique identifier. After deletion of the element, no other element will receive the same Puid.

3.2.6.14 element Trade_t/Trade

Description	
Type	**knx:Trade_t**

3.2.6.15 element Trade_t/DeviceInstanceRef

Description	References a device contained in a trade.
Type	**knx:DeviceInstanceRef_t**

3.2.7 Group Addresses

3.2.7.1 element Project_t/Installations/Installation/GroupAddresses

Description	Contains the group address structure
Type	**knx:GroupAddresses_t**

3.2.7.2 complexType GroupAddresses_t

Description	Contains the group address structure
Children	Name Description **GroupRange** List of named group address ranges

3.2.7.3 element GroupRange_t/GroupAddress

Description	Describes a group address				
Attributes	Name	Type	Use	Default	Description
	Id	xs:ID	required		Unique ID.

					On export or conversion, this will be constructed as *parid*_**GA-***number*, where: *parid* ID of the parent Project and InstallationID sepearted with '-' *number* Unique number of the group addess within the project. This does not reflect the address value! For converted projects, this corresponds to GroupAddress.UniqueNumber in the database schema.
	Address	xs:unsignedInt	required		Group address [1…65535]
	Name	knx:String255_t	required		Name
	Unfiltered	xs:boolean	optional	false	If true, the group addresses in the range will not be filtered by routers. Note that if a group address is part of one or more GroupRanges with Unfiltered=true, it will not be filtered irrespective of the setting of Unfiltered in the GroupAddress.
	Central	xs:boolean	optional	false	If true, the group address will be treated as central address during copy operations.
	Global	xs:boolean	optional	false	If true, the group address will be used in all installations of the project. Global groups must have the same address and type in all installations of a project.
	Description	xs:string	optional		Description
	Comment	xs:string	optional		Comment
	DatapointType	knx:IDREF	optional		Optional datapoint type specification. A reference to DatapointType or DatapointSubtype. If the group address is linked to any DeviceCommunicationObjects, the sizes must match.
	Puid	xs:string	required		The project wide unique identifier. After deletion of the element, no other element will receive the same Puid.
	Key	knx:Aes128_t	optional		The key used for data security communication. All senders and receivers of this group address use the same key.
	Security	knx:SecurityMode	optional	Auto	Defines the security mode for the group address. Can be either Auto, On or Off.

3.2.7.4 element GroupAddresses_t/GroupRanges/GroupRange

Description	Top-level named group range
Type	extension of **knx:GroupRange_t**

3.2.7.5 complexType GroupRange_t

Description	Element of the group address structure				
Children	Name	Description			
	GroupRange	Child group ranges			
	GroupAddress	GroupAddresses located within the GroupRange			
Attributes	Name	Type	Use	Default	Description
	Id	xs:ID	required		Unique ID. On export or conversion, this will be constructed as *parid*_**GR-***number*, where: *parid* ID of the parent Project and InstallationID sepearted with '-' *number* Unique number of the group range within the project.
	Name	knx:String255_t	required		Name

	RangeStart	xs:unsignedShort	required		First possible group address in the range
	RangeEnd	xs:unsignedShort	required		Last possible group address in the range
	Unfiltered	xs:boolean	optional	false	If true, all group addresses in the range will not be filtered by routers; irrespective of the individual setting of GroupAddress/@Unfiltered.
	Description	xs:string	optional		Description
	Comment	xs:string	optional		Comment
	Puid	xs:string	required		The project wide unique identifier. After deletion of the element, no other element will receive the same Puid.
	Security	knx:SecurityMode	optional	Auto	Defines the security mode for the group addresses within the range or any child range.

3.2.7.6 element GroupRange_t/GroupRange

Description	Child named group address range
Type	extension of **knx:GroupRange_t**

3.2.8 SplitInfos

3.2.8.1 element Project_t/Installations/Installation/SplitInfos

Description	The required information about a split installation..
Type	**knx:SplitInfos_t**

3.2.8.2 complextType SplitInfos_t

Description	Collection of SplitInfo elements, used for Split & Merge
Type	extension of **knx:SplitInfo_t**
Children	Name Description **SplitInfo** Any number of split infos

3.2.8.3 element SplitInfo_t/SplitInfo

Description	The required information about a split installation..
Type	**knx:SplitInfo_t**

3.2.8.4 complexType SplitInfo_t

Description	An element with information for Split & Merge
Attributes	Name Type Use Default Description ObjectPath xs:string required Cookie xs:string required Pattern for the cookie: [0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12}

4 IDs and relations

4.1 ID naming schema

This section summarizes the naming rules for elements of the KNX XML schema. All these IDs are constructed so that they are globally unique. Detailed descriptions are included in the individual element descriptions. Note that many IDs of subordinate elements start with the ID of the parent element, then – separated by an underscore – additional specification.

Often part of the constructed ID is a unique number. How this number is to be generated and which unique constraints apply for the given element is described in detail in the individual element descriptions.

Because IDs can contain only letters, digits, dot, hyphen and underscore characters (see XML Namespaces specification, production for NCName), and hyphen and underscore are already used as separators, all characters from strings that are not letters or digits have to be escaped: A character which is neither a letter nor a digit is represented as a dot, followed by 2 hexadecimal digits representing the UTF-8 encoding of the character. Example: a slash (/) is represented as ".2F", a German umlaut ä (Unicode code point U+00E4) as ".C3.A4".

4.1.1 MasterData

Element Type	ID Naming	Example
DatapointType	<DatapointType@Id> ::= **DPT-** <DatapointType@Number>	"DPT-15"
DatapointSubtype	<DatapointSubtype@Id> ::= **DPST-** <DatapointType@Number> **-** <DatapointSubtype@Number>	"DPST-15-0"
MediumType	<MediumType@Id> ::= **MT-** <MediumType@Number>	"MT-1"
Manufacturer	<Manufacturer@Id> ::= **M-** <Manufacturer@KnxManufacturerId:X4>	"M-0001"

4.1.2 Manufacturer Data

Element Type	ID Naming	Example
ApplicationProgram	<ApplicationProgram@Id> ::= <Manufacturer@Id> **_A-** <@ApplicationNumber:X4> **-** <@ApplicationVersion:X2>-<HashPart:X4> | <Manufacturer@Id> **_A-** <@ApplicationNumber:X4> **-** <@ApplicationVersion:X2>-<HashPart:X4> **-O** <@OriginalManufacturer>	"M-0001_A-2419-01-BAF8"
Parameter	<Parameter@Id> ::= <ApplicationProgram@Id> **_P-** UniqueNumber()	"M-0001_A-2419-01-BAF8_P-107"
UnionParameter	<UnionParameter@Id> ::= <ApplicationProgram@Id> **_UP-** UniqueNumber()	"M-0001_A-2419-01-BAF8_UP-111"
ParameterRef	<ParameterRef@Id> ::= <ParameterRef@RefId> **_P-** UniqueNumber()	"M-0001_A-2419-01-BAF8_P-9_R-79"
ParameterCalculation	<ParameterCalculation@Id> ::= <ApplicationProgram@Id> **_PC-** UniqueNumber()	"M-0001_A-902B-12-F67A_PC-1"
ComObject	<ComObject@Id> ::= <ApplicationProgram@Id> **_O-** UniqueNumber()	"M-0001_A-2419-01-BAF8_O-7"
ComObjectRef	<ComObjectRef@Id> ::= <ComObjectRef@RefId> **_R-** UniqueNumber()	"M-0001_A-2419-01-BAF8_O-5_R-72"
BinaryData	<BinaryData@Id> ::= <ApplicationProgram@Id> **_BD-** <BinaryData@Name>	
Hardware	<Hardware@Id> ::= <Manufacturer@Id> **_H-** <Hardware@SerialNumber> **-** <Hardware@VersionNumber>[**-O** <@OriginalManufacturer>]	"M-0001_H-hp.5F00010-1"
Hardware2Program	<Hardware2Program@Id> ::= <Hardware@Id> **_HP** | <Hardware@Id> **_HP-** <@ApplicationNumber:X4> **-** <@ApplicationVersion:X2>-<HashPart:X4> [**-O** <@OriginalManufacturer>] | <Hardware@Id> **_HP-** <@ApplicationNumber:X4> **-** <@ApplicationVersion:X2>-<HashPart:X4> [**-O** <@OriginalManufacturer>]	"M-0001_H-hp.5F00181-1_HP" "M-0001_H-hp.5F00105-1_HP-9010-02-842D" "M-0001_H-hp.5F00185-1_HP-8023-11-AB36-0053-01-48F3"

	- <@ApplicationNumber:X4> **-** <@ApplicationVersion:X2>-<HashPart:X4> [**-O** <@OriginalManufacturer>]	
Product	<Product@Id> ::= <Hardware@Id> **_P-** <Product@OderNumber>	"M-0001_H-hp.5F00185-1_P-5WG1.20141.2D4AB01.20.20"

4.1.3 Project Data

Element Type	**ID Naming**	**Example**
Project	<Project@Id> ::= **P-** UniqueNumber():X4	"P-3AD2"
Area	<Area@Id> ::= <Project@Id> - <Installation@InstallationId> **_A-** UniqueNumber()	"P-3AD2-1_A-3"
Line	<Line@Id> ::= <Project@Id> - <Installation@InstallationId> **_L-** UniqueNumber()	"P-3AD2-1_L-57"
BuildingPart	<BuildingPart@Id> ::= <Project@Id> - <Installation@InstallationId> **_BP-** UniqueNumber()	"P-3AD2-1_BP-3"
Function	<Function@Id> ::= <Project@Id> - <Installation@InstallationId> **_F-**UniqueNumber()	"P-3AD2-1_F-3"
GroupAddressRef	<GroupAddressRef@Id> ::= <Project@Id> - <Installation@InstallationId> **_GF-**UniqueNumber()	"P-3AD2-1_GF-89"
Trade	<Trade@Id> ::= <Project@Id> - <Installation@InstallationId> **_T-** UniqueNumber()	"P-3AD2-1_T-1"
DeviceInstance	<DeviceInstance@Id> ::= <Project@Id> - <Installation@InstallationId> **_DI-** UniqueNumber()	"P-3AD2-1_DI-3"
GroupRange	<GroupRange@Id> ::= <Project@Id> - <Installation@InstallationId> **_GR-** UniqueNumber()	"P-3AD2-1_GR-1"
GroupAddress	<GroupAddress@Id> ::= <Project@Id> - <Installation@InstallationId> **_GA-** UniqueNumber()	"P-3AD2-1_GA-1"

4.2 Reference Summary

This section summarizes the IDREF – ID relations between the elements. The last column contains an X if the referenced ID may be in another XML file.

4.2.1 Manufacturer Data → Manufacturer Data

From	Attribute	To	
ApplicationProgram_t	OriginalManufacturer	Manufacturer	X
Parameter_t	ParameterType	ParameterType	
ComObjectRef_t	RefId	ComObject	
BinaryDataRef_t	RefId	BinaryData	
Hardware_t	OriginalManufacturer	Manufacturer	X

4.2.2 Project Data → Master Data

From	Attribute	To	
Topology_t/Area/Line	MediumTypeRefId	MediumType	X
ComObjectInstanceRef_t	DatapointType	DatapointType or DatapointSubtype	X
GroupAddresses_t/GroupAddress/DatapointType	DatapointType	DatapointType or DatapointSubtype	X

4.2.3 Project Data → Manufacturer Data

From	Attribute	To	
DeviceInstance_t	ProductRefId	Product	X
DeviceInstance_t	Hardware2ProgramRefId	Hardware2Program	X
DeviceInstance_t/ParameterInstanceRefs/ParameterInstanceRef	RefId	ParameterRef	X
ComObjectInstanceRef_t	RefId	ComObjectRef	X
DeviceInstance_t/BinaryData/BinaryData	RefId	BinaryData	X

4.2.4 Project Data → Project Data

From	Attribute	To	
Topology_t/Area/Line/AdditionalGroupAddresses/GroupAddressRef	RefId	GroupAddress	
BuildingPart_t/Function/GroupAddressRef	RefId	GroupAddress	
ComObjectInstanceRef_t/Connectors/Send	GroupAddressRefId	GroupAddress	
ComObjectInstanceRef_t/Connectors/Receive	GroupAddressRefId	GroupAddress	
DeviceInstanceRef_t	RefId	DeviceInstance	

5 Transfer files

For export and import scenarios, the generated XML file(s) packed into a ZIP archive. This has the following advantages:

- *By compression, the files have a manageable size*
- *Not everything needs to be in a single XML*

This is important since current XML parsers and *XPath* implementations do not work well or do not work at all on huge XML files. The knx:IDREF need not resolve within each individual XML file within the archive, but within the whole archive. For import, the individual XML files may also be present unzipped, but in the same file system directory.

5.1 File extensions

As file extension, the following is used:

*.knxprod	If just master and manufacturer product data is included
*.knxproj	If master, product and project data is included.

5.2 Content

5.2.1 Non-XML files

The following data is not stored within the XML files but as external files

- Baggage data
- BinaryData and BinaryDataRef data within device instance data
- UserFile data

The corresponding XML elements omit the Data child element.

5.2.2 Distribution to partial XML files

When distributing the data to different XML files, the following restrictions apply:

- All MasterData is in one XML file.
- Together with an ApplicationProgram element, all child elements must be in the same XML file.
- Together with a Project element, all child elements must be in the same XML file.

Logically, the files can be thought of as a merged XML file. In principle, starting from the KNX element, the files are merged recursively, with the following rules:

- The following elements will be identified (within a recursion level); they must have identical attributes in each partial XML.
    - Elements with same tag and same "Id"
    - Elements with same tag without "Id" (this is for the container-type elements like e.g. "ManufacturerData").
    - Language elements with same "Identifier"
    - Language/Translation elements with same "RefId"
    - Language/Translation/Translation elements with same "AttributeName"
    - Exception: Project is never merged (two projects even with the same name will stay two distinct projects)
    - Below ApplicationProgram no merging is required; here everything must be identical.

The converter will produce the partial XML files according to the following rules:

- Each ApplicationProgram element will be written to a separate XML file
- Each Baggage element will be written to a separate XML file
- Each Project element will be written to a separate XML file

5.2.3 Naming convention

To avoid name conflicts between the individual XML files within the archive, the following naming convention should be obeyed:

knx_master.xml	Created by KNX; contains only master data.
M-*iiii*/Baggages.xml	Created by manufacturer *iiii* (manufacturer ID, formatted as 4 hex digits); contains baggage data.
M-*iiii*/Catalog.xml	Created by manufacturer *iiii* (manufacturer ID, formatted as 4 hex digits); contains catalog data.
M-*iiii*/Hardware.xml	Created by manufacturer *iiii* (manufacturer ID, formatted as 4 hex digits); contains hardware data.
M-*iiii*/M-*iiii*_A-*nnnn*-*vv*-*ffff*.xml	Created by manufacturer *iiii* (manufacturer ID, formatted as 4 hex digits); contains the data for the application program *nnnn* in version *vv* with fingerprint *ffff*.
P-*iiii*/project.xml	Created by user; contains the global data for project *iiii* (internal project ID, formatted as 4 hex digits).
P-*iiii*/*n*.xml	Created by user; contains the data for installation *n* of project *iiii* (internal project ID, formatted as 4 hex digits).
*.xml	Created by user; contains project data (* should not contain – and _ characters).

5.2.4 Password protection

When exporting a password-protected project, the proj_*.xml file may optionally be ZIP encoded with the project password.

Note that there is no way to recover or reset a lost ZIP password!

5.3 ETS Container Structure

The converter creates *.knxprod files containing the ETS4 product data and *.knxproj files containing ETS project data. Both file formats are renamed zip files that contain several xml files following the KNX-XML schema.

5.3.1 ETS Product Structure

ETS4 uses for project the extension *.knxprod; the container contains the following files:

- The root of the zip container contains one file, the knx_master.xml, which contains all KnxMasterData.
- For every manufacturer, a subfolder is created, to which all files from that manufacturer are written. The name of the folder is the <Manufacturer.RefId> (e.g. "M-0001").
- A single file is written for each ApplicationProgram element. This file is located in the manufacturer folder. The name of the ApplicationProgram file is "<ApplicationProgramId>.xml" (e.g. “M-0001_A-0002-21-25A6.xml”).This xml file not only contains the application program element with all its child elements but also all the translation units referencing this application program element.
- The data from Manufacturer\Catalog is written to "Catalog.xml" and is also located in the manufacturer folder, the catalog belongs to. Similar to the application program XML this file also contains all the translation units referencing translations for catalog sections and catalog items.
- All hardware data is written to "Hardware.xml", also located in the manufacturer folder. Again this file contains all the translation units referencing hardware product element translations.
- If at least one baggage from the current manufacturer exists, a subfolder named "Baggages" is created and the information for the baggages is written to "Baggages.xml". The baggage data itself is not included in this file but are stored in the "Baggages" subfolder as separate files, according to their TargetPath and Name. (e.g. for a baggage with TargetPath = "TMw" and Name = "010_TMwPlugIn_0407.chm", the baggage data is stored in the file "\M-0001\Baggages\TMw\010_TMwPlugIn_0407.chm"
- To ensure integrity of all the product data in the different manufacturer folders, each folder is hashed/ signed in an external signature file named <FolderName>.signature. This signature file is located in the root of the zip container. Without a valid signature file it is impossible to import product data from the corresponding manufacturer’s folder.

Example of folder and file structure for a *.knxprod container (not all baggage's are listed):

---\knx_master.xml
---\M-0002.signature
---\M-0002\Catalog.xml
---\M-0002\Hardware.xml
---\M-0002\Baggages.xml
---\M-0002\M-0002_A-A00E-16-98A2.xml
---\M-0002\Baggages\ABB_RC01PlugIn0407.loc
---\M-0002\Baggages\RC010022\RC010409.TXT

5.3.2 ETS Project Structure

ETS uses for project the extension *.knxprod ; the container contains the following files:

- For every used manufacturer in project (means its devices) the ***ETS Product Structure*** section as described above
- For the project folder (e.g. P-3D5F),
    - The project folder hashed/ signed by ETS4 on export in an external signature file named <FolderName>.signature. This signature file is located in the root of the zip container.
    - The <FolderName>\Project.xml contains project organizational data (under the Project ID)
        - ProjectInformation
        - HistoryEntries
        - ProjectTraces
        - UserFiles
    - The <FolderName>\0.xml contains the project topology (under the Project ID)
        - <number>.xml → 0…16 is the preparation for “installations” with up to 16 projects, currently not used

Example of folder and file structure for a *.knxproj container:

---\knx_master.xml
---\M-0001.signature
---\M-0002.signature
---\P-3D5F.certificate
---\P-3D5F.signature
:
:
---\P-3D5F\Project.xml
---\P-3D5F\0.xml
---\P-3D5F\UserFiles\887190.txt

5.3.3 Password protected projects

ETS5, like ETS3/4, allows password protection for ETS project data. Due to the new format of persisted data, the mechanism for password protection in ETS4/5 differs from ETS3.

Password-protected projects in older ETS versions

In older ETS versions (ETS1, ETS2, ETS3), password-protection of project data was merely an internal property, and did not affect the format of exported file data (*.prx, *.pr1, *.pr2, *.pr3, *.pr4, *.pr5). This was not necessary, as the file format was cryptic anyway.

The password became relevant only when a user wanted to open a project inside the UI. On password protection, the ETS UI then asked for the password before letting the user view or edit the data of the project in question.

Password-protected projects in ETS4/5

ETS4/5 however uses a completely different persistence format. Project data are persisted as *.knxproj files. Each *.knxproj file is an ordinary unprotected ZIP archive, which may contain various XML files, subfolders, and possibly additional baggage files. In particular, project data in the narrower sense normally reside in a separate subfolders named "P-*".

For password-protected projects, the files normally contained in the "P-*" subfolder are put in an extra, password-protected ZIP-file named "P-*.zip", which then replaces the subfolder "P-*" of this project.

Example

The following example illustrates schematically the difference between the contents of a *.knxproj file with and without password protection, respectively.

Without password	**With password**	**Comment**
knx_master.xml	knx_master.xml	*KNX master data*
M-000B\Hardware.xml M-000B\M-000B_A-1151-10-12C6.xml	M-000B\Hardware.xml M-000B\M-000B_A-1151-10-12C6.xml	*Manufacturer-specific data*
P-01A0\0.xml **P-01A0\project.xml**	**P-01A0.zip** (password-protected)	*Project-specific data*

Export

When exporting a password-protected project, the proj_*.xml file may optionally be ZIP encoded with the project password.

Note that there is no way to recover or reset a lost ZIP password!