



# The OpenVX<sup>™</sup> XML Schema Extension

The Khronos<sup>®</sup> OpenVX Working Group, Editors: Jesse Villarreal, Erik Rainey,  
Radhakrishna Giduthuri

Version 1.1 (provisional), Mon, 10 Dec 2018 23:43:00 +0000

# Table of Contents

1. XML Schema Extension .....	2
1.1. Purpose .....	2
1.2. Motivation .....	2
1.3. Schema .....	2
2. Module Documentation .....	3
2.1. Extension: XML API .....	3
2.1.1. Typedefs .....	3
2.1.2. Enumerations .....	3
2.1.3. Functions .....	4



Copyright 2013-2018 The Khronos Group Inc.

This specification is protected by copyright laws and contains material proprietary to Khronos. Except as described by these terms, it or any components may not be reproduced, republished, distributed, transmitted, displayed, broadcast or otherwise exploited in any manner without the express prior written permission of Khronos.

This specification has been created under the Khronos Intellectual Property Rights Policy, which is Attachment A of the Khronos Group Membership Agreement available at [www.khronos.org/files/member\\_agreement.pdf](http://www.khronos.org/files/member_agreement.pdf). Khronos Group grants a conditional copyright license to use and reproduce the unmodified specification for any purpose, without fee or royalty, EXCEPT no licenses to any patent, trademark or other intellectual property rights are granted under these terms. Parties desiring to implement the specification and make use of Khronos trademarks in relation to that implementation, and receive reciprocal patent license protection under the Khronos IP Policy must become Adopters and confirm the implementation as conformant under the process defined by Khronos for this specification; see <https://www.khronos.org/adopters>.

Khronos makes no, and expressly disclaims any, representations or warranties, express or implied, regarding this specification, including, without limitation: merchantability, fitness for a particular purpose, non-infringement of any intellectual property, correctness, accuracy, completeness, timeliness, and reliability. Under no circumstances will Khronos, or any of its Promoters, Contributors or Members, or their respective partners, officers, directors, employees, agents or representatives be liable for any damages, whether direct, indirect, special or consequential damages for lost revenues, lost profits, or otherwise, arising from or in connection with these materials.

Khronos is a registered trademark, and OpenVX is a trademark of The Khronos Group Inc. OpenCL is a trademark of Apple Inc., used under license by Khronos. All other product names, trademarks, and/or company names are used solely for identification and belong to their respective owners.

### **Contributors**

- Erik Rainey, Amazon
- Jesse Villarreal, Texas Instruments

# Chapter 1. XML Schema Extension

## 1.1. Purpose

The purpose of this extension is to create and standardize a description of an OpenVX Context (a set of graphs and their related data objects) in XML format.

## 1.2. Motivation

The intent is to standardize a representation of the OpenVX Context with an XML Schema and to standardize on an API. Having a standardized Schema means that:

- Some amount of validation for OpenVX Graph Verification may happen at *Import* time.
- Graphs and data may now be platform *portable*.
- The XML may be parsed or created by external tools for a variety of uses:
  - Documentation
  - Standards Compliance
  - Language Portability

## 1.3. Schema

The XML Schema for a specific version of OpenVX is available at the following url folder:

- <https://www.khronos.org/registry/vx/schema/>

The name of the schema file is of the format: **openvx-VERSION.xsd** , where VERSION, is the OpenVX version number, with hyphens separating the major and minor version numbers.

# Chapter 2. Module Documentation

## 2.1. Extension: XML API

The Khronos Extension for OpenVX XML Import and Export Support.

### Typedefs

- [vx\\_import](#)

### Enumerations

- [vx\\_ext\\_import\\_type\\_e](#)
- [vx\\_ext\\_import\\_types\\_e](#)
- [vx\\_import\\_attribute\\_e](#)

### Functions

- [vxExportToXML](#)
- [vxGetImportReferenceByIndex](#)
- [vxGetImportReferenceByName](#)
- [vxImportFromXML](#)
- [vxQueryImport](#)
- [vxReleaseImport](#)

### 2.1.1. Typedefs

#### **vx\_import**

An abstract handle to an import object.

```
typedef struct _vx_import *vx_import;
```

### 2.1.2. Enumerations

#### **vx\_ext\_import\_type\_e**

The Object Type Enumeration for Imports.

```
enum vx_ext_import_type_e {  
    VX_TYPE_IMPORT = 0x814,  
};
```

#### **Enumerator**

- **VX\_TYPE\_IMPORT** - A [vx\\_import](#)

## **vx\_ext\_import\_types\_e**

The import type enumeration.

```
enum vx_ext_import_types_e {  
    VX_IMPORT_TYPE_XML = 0,  
};
```

See also: [VX\\_IMPORT\\_ATTRIBUTE\\_TYPE](#)

### **Enumerator**

- **VX\_IMPORT\_TYPE\_XML** - The XML import type.

## **vx\_import\_attribute\_e**

The import attributes list.

```
enum vx_import_attribute_e {  
    VX_IMPORT_ATTRIBUTE_COUNT = VX_ATTRIBUTE_BASE(VX_ID_KHRONOS, VX_TYPE_IMPORT) +  
    0x0,  
    VX_IMPORT_ATTRIBUTE_TYPE = VX_ATTRIBUTE_BASE(VX_ID_KHRONOS, VX_TYPE_IMPORT) + 0x1,  
};
```

See also: [vxQueryImport](#)

### **Enumerator**

- **VX\_IMPORT\_ATTRIBUTE\_COUNT** - Returns the number of references in the import object. Use a **vx\_uint32** parameter.
- **VX\_IMPORT\_ATTRIBUTE\_TYPE** - Returns the type of import. Use a **vx\_ext\_import\_types\_e** parameter.

## **2.1.3. Functions**

### **vxExportToXML**

Exports all objects in the context to an XML file which uses the OpenVX XML Schema.

```
vx_status vxExportToXML(  
    vx_context          context,  
    vx_char             xmlfile[]);
```

### **Parameters**

- **[in]** *context* - The context to export.
- **[in]** *xmlfile* - The file name to write the XML into.



*Note*

The reference numbers contained in the xml file can appear in any order but should be inclusive from index number 0 to [number of references - 1]. For example, if there are 20 references in the xml file, none of the reference indices should be  $\geq 20$ .

**Returns:** A `vx_status_e` enumeration.

**See also:** <https://www.khronos.org/registry/vx/schema/opencvx-1-1.xsd>

### **vxGetImportReferenceByIndex**

Used to retrieve a reference by the index from the import.

```
vx_reference vxGetImportReferenceByIndex(  
    vx_import import,  
    vx_uint32 index);
```

#### **Parameters**

- `[in]` *import* - The reference to the import object.
- `[in]` *index* - The index of the reference in the import object to return.

**Returns:** `vx_reference`

#### **Return Values**

- 0 - Invalid import object or index.
- \* - The reference at the requested index number.



*Note*

Use `vxQueryImport` with `VX_IMPORT_ATTRIBUTE_COUNT` to retrieve the upper limit of references in the import.



*Note*

Use `vxReleaseReference` to release the reference before releasing the context.

**Precondition:** `vxImportFromXML`

### **vxGetImportReferenceByName**

Used to retrieve a reference by name from the import when the name is known beforehand. If multiple references have the same name, then *any* one of them may be returned.

```
vx_reference vxGetImportReferenceByName(  
    vx_import import,  
    const vx_char* name);
```

### Parameters

- **[in]** *import* - The reference to the import object.
- **[in]** *name* - The reference string name.

**Returns:** *vx\_reference*

### Return Values

- 0 - Invalid import object or name does not match a reference in the import object.
- \* - The reference matching the requested name.



#### Note

Use [vxReleaseReference](#) to release the reference before releasing the context.

**Precondition:** [vxImportFromXML](#)

### **vxImportFromXML**

Imports all framework and data objects from an XML file into the given context.

```
vx_import vxImportFromXML(  
    vx_context context,  
    vx_char xmlfile[]);
```

### Parameters

- **[in]** *context* - The context to import into.
- **[in]** *xmlfile* - The XML file to read.



#### Note

The reference indices in the import object corresponds with the reference numbers in the XML file. It is assumed that the program has some means to know which references to use from imported list (either by name: [vxGetImportReferenceByName](#), or by index from looking at the XML file (debug use case): [vxGetImportReferenceByIndex](#)). Alternatively, the program can use [vxGetImportReferenceByIndex](#) in a loop and query each one to understand what was imported. After all references of interest have been retrieved, this import objects should be released using [vxReleaseImport](#).

**Returns:** *vx\_import* object containing references to the imported objects in the context



See also: <https://www.khronos.org/registry/vx/schema/opencvx-1-1.xsd>

## vxQueryImport

Used to query the import about its properties.

```
vx_status vxQueryImport(  
    vx_import          import,  
    vx_enum            attribute,  
    void*              ptr,  
    vx_size            size);
```

### Parameters

- **[in]** *import* - The reference to the import object.
- **[in]** *attribute* - The `vx_import_attribute_e` value to query for.
- **[out]** *ptr* - The location at which the resulting value will be stored.
- **[in]** *size* - The size of the container to which *ptr* points.

**Returns:** A `vx_status_e` enumeration.

**Precondition:** `vxImportFromXML`

## vxReleaseImport

Releases a reference to an import object. Also internally releases its references to its imported objects. These imported objects may not be garbage collected until their total reference counts are zero.

```
vx_status vxReleaseImport(  
    vx_import*          import);
```

### Parameters

- **[in]** *import* - The pointer to the import object to release.

**Returns:** A `vx_status_e` enumeration.

### Return Values

- `VX_SUCCESS` - No errors.
- `VX_ERROR_INVALID_REFERENCE` - If *import* is not a `vx_import`.



#### Note

After returning from this function the reference will be zeroed.

**Precondition:** `vxImportFromXML`