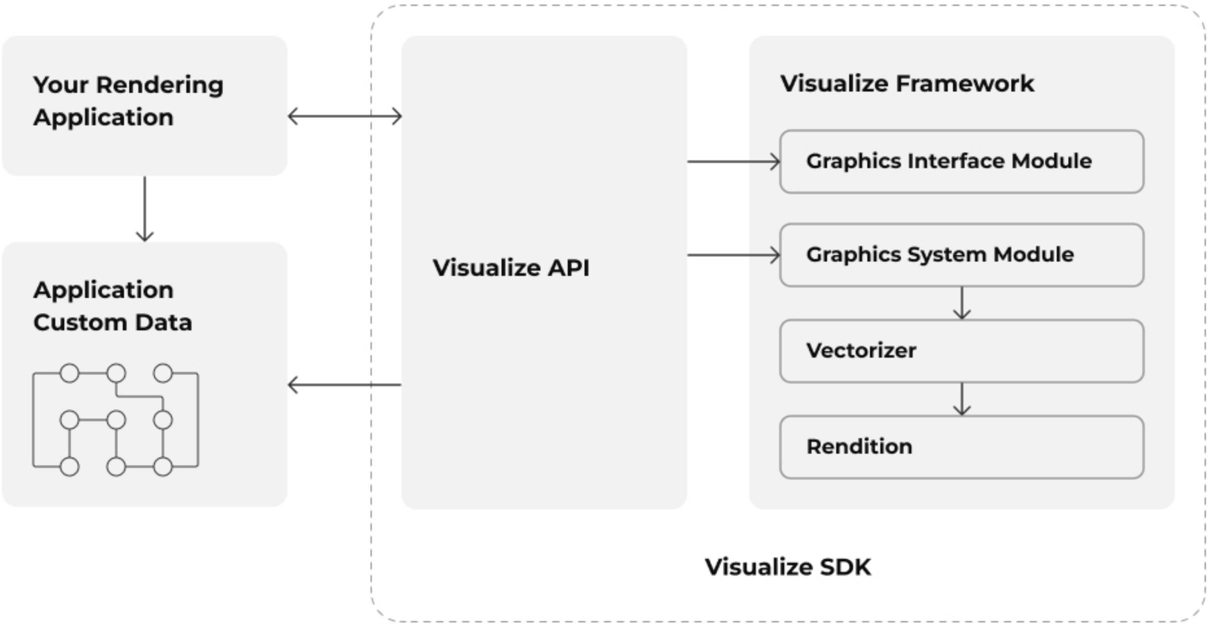
Why Use Visualize SDK? Use Cases

Visualize is a C++ SDK offering advanced visualization and seamless integration with other ODA products. The inWEB version provides JS wrappers over the C++ code.

Visualize SDK enables cross-platform applications for visualizing CAD and BIM formats, delivering high-quality rendering with minimal effort and database independence.

Visualize SDK

Visualize SDK combines Visualize Framework and Visualize API to provide a powerful and handy toolset for rendering data of any file format. The diagram below shows an interaction between a rendering application and Visualize SDK:



Visualize Framework

It consists of the Graphics Interface (GI) Module, Graphics System (GS) Module, Vectorizer and Rendition. The GS module provides an API for loading Vectorization Modules and managing views and devices. The GI module's API loads geometry, which is then passed to the Vectorizer for actions like calculating attributes and simplifying geometry. Finally, the data is sent to the Rendition for output.

Visualize API

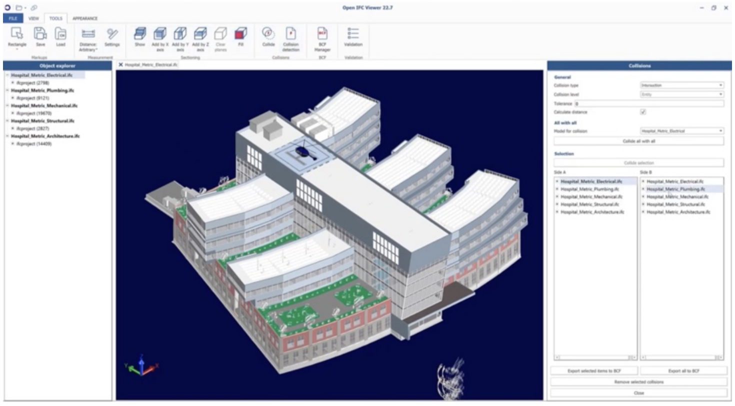
Visualize API provides access to the Visualize Framework's advanced rendering features, independent of database types. The core of the SDK is the Internal Universal Database, storing geometry primitives like circles and polylines, managed via the GI and GS APIs.

Visualize Features

docs.opendesign.com

Let’s review some of the most interesting advanced features available:

- **Collision Detection –** Link to the video

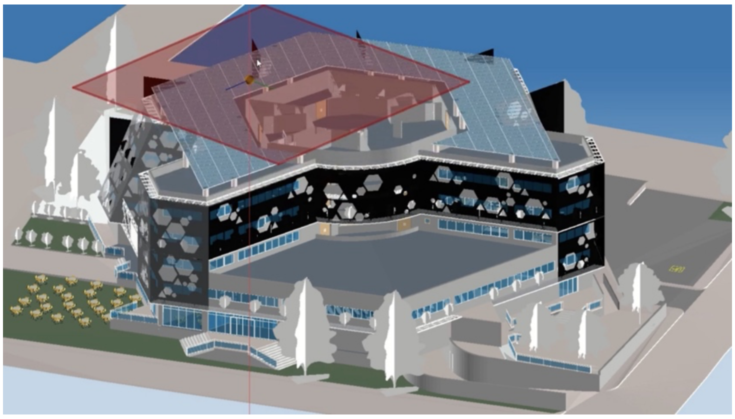


Collision type: Intersection
Collision level: Entity
Tolerance: 0
Calculate distance: ☑
Model for collision: Hospital_Metric_Electrical

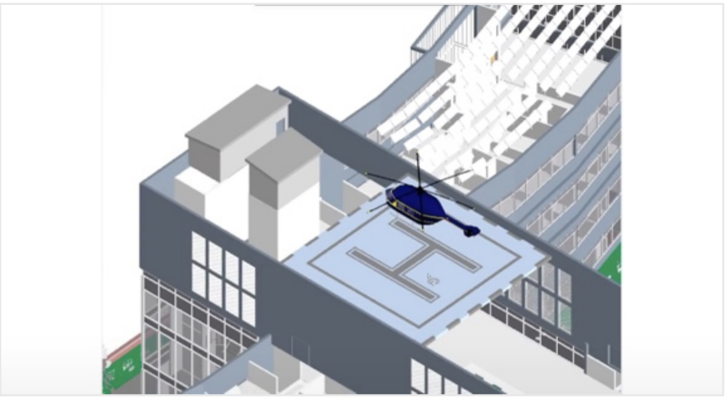
Side A:
- Hospital_Metric_Electrical.ifc
- Hospital_Metric_Plumbing.ifc
- Hospital_Metric_Mechanical.ifc
- Hospital_Metric_Structural.ifc
- Hospital_Metric_Architecture.ifc

Side B:
- Hospital_Metric_Electrical.ifc
- Hospital_Metric_Plumbing.ifc
- Hospital_Metric_Mechanical.ifc
- Hospital_Metric_Structural.ifc
- Hospital_Metric_Architecture.ifc

- **Cutting Planes** – Link to the video



- **Animation –** Link to the video



- **Selections**
- **Highlighting**

Visualize allows you to implement animation for entities, sub-entities and cameras. The Animation API has three possible transformations: translation, rotation and scaling.

Partial Viewing

Partial viewing is a feature that allows you to partially load and view database objects with which you are currently working. This means that only objects that are currently viewed (or seen by a camera) are loaded and rendered. The rest of the objects remain unloaded until they are viewed by an active view. This can help to work with large files, specifically when you need to work only with a portion of a file.

Manage Display of a Model Content

Most of the time, models have several graphical objects, and sometimes it is useful to hide or isolate them (make an object the single visible object in a scene). You can hide or isolate model content according to the selection, or perform these actions with a specific entity or geometry data.

Ambient Occlusion Effect

It is a rendering technique that simulates global illumination by adding shading to geometry corners, making the scene look more realistic by decreasing light intensity where it re-emits multiple times.

Anti-Aliasing

Geometry looks smoother and visually more correct when anti-aliasing is enabled. Visualize supports two types of anti-aliasing: anti-aliasing for linear geometries, like polylines, arcs, NURBS curves, and FXAA (Fast Approximation Anti-Aliasing).

User Data Management

User data (also known as XData, extended data, or custom data) is a data set that is used by an external application; it can be stored in the database together with an object to which the data belongs.

Common Data Access (CDA)

The Common Data Access (CDA) mechanism is used by various applications (e.g., OpenIfcViewer, ODA Viewer, etc.) to access structural information and object properties of imported (open) databases in an abstract manner without the need to know about the imported format. The structure is provided as a tree, where the nodes of the tree are connected with imported instances. Specific levels of the hierarchy in the tree depend on the domain, as well as the connection of this perceived hierarchy with the database content.

Visualize Database – VSFX

VSFX (VSF is the old version) is a file format created by our Visualize team. It saves the visualize database with all models, their geometry, devices, views, and more. The ODA Platform supports various data formats such as DWG including extensions for Map, Architecture, Civil and Mechanical, DGN, DXF, RVT/RFA, NWC, IFC,STEP. If you use the format SDKs provided by ODA, you benefit from seamless conversion to VSFX. For visualizing data formats outside the ODA Platform, you would need to develop a converter to transform those formats into VSFX.

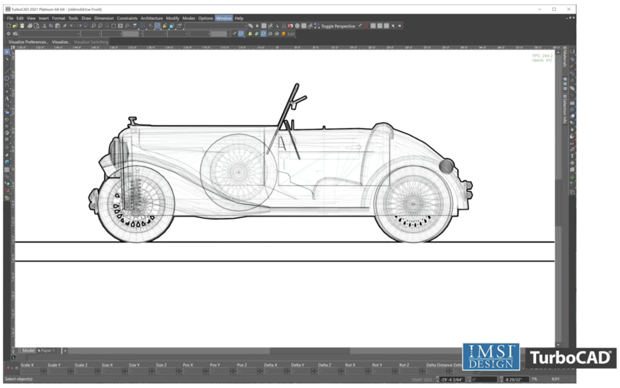
docs.opendesign.com

Most Visualize objects are contained in a database (objects are database-residents). There are two exceptions: the database object itself and the factory from which a database is created. All Visualize objects can be created using a database or methods of its resident objects.

Visualize database can store

- Models
- Blocks
- Linetypes

- Layers
- Text styles
- Materials
- Raster images
- Devices (vectorization modules)
- Visual styles – Link to the video



- Highlight styles
- Backgrounds

The general database structure is shown below.

Sample Applications

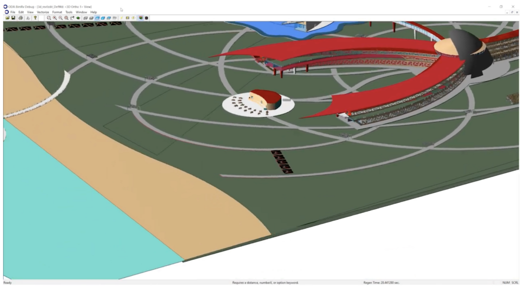
Here you can find an overview of the Visualize Sample applications available to all ODA Members, including trial users. If you're interested in any of these, links to the documentation will become available immediately after applying for a free trial.

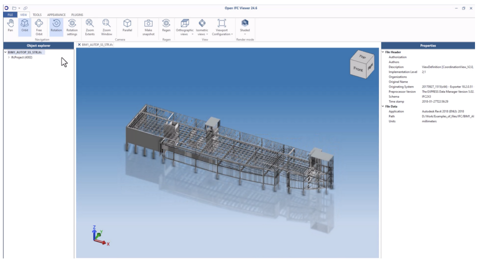
Acis2Visualize	Sample application (.tx module) that demonstrates importing ACIS models to Visualize universal database and subsequent rendering.
ODAViewer	Source files and assets for the ODA Viewer sample application that can run on Android OS
Obj2Visualize	Sample application (.tx module) that demonstrates importing .obj files to Visualize universal database and subsequent rendering.
PointCloud2Visualize	Sample application (.tx module) that demonstrates importing .pts and .xyz files to Visualize universal database and subsequent rendering.
Rcs2Visualize	Sample application (.tx module) that demonstrates importing a .rcs file to Visualize universal database and subsequent rendering.
Stl2Visualize	Sample application (.tx module) that shows an example of importing .stl files to Visualize universal database and subsequent rendering.
Visualize2Obj	Sample application (.tx module) that demonstrates exporting 3D geometry that is open in a Visualize application to OBJ file format.
VisualizeModelsGenerator	A sample that generates model content. The sample is built into a dynamic module (VisualizeModelsGenerator_*.tx file) that can be loaded and used in any application hosted by ODA Platform (for example, in the Visualize Viewer sample application).

Visualize Viewer	Sample GUI application that provides a set of Qt widgets used to create, modify and render Visualize Stream Files (.vsf) and many other file formats.
Bim2Visualize	Sample application (.tx module) that demonstrates importing .rvt and .rfa files to Visualize universal database and subsequent rendering.
Dgn2Visualize	Sample application (.tx module) that demonstrates importing .dgn drawings to Visualize universal database and subsequent rendering.
Dwf2Visualize	Sample application (.tx module) that demonstrates importing .dwf and dwfx files to Visualize universal database and subsequent rendering.
Dwg2Visualize	Sample application (.tx module) that demonstrates importing .dwg files to Visualize universal database and subsequent rendering.
Ifc2Visualize	Sample application (.tx module) that demonstrates importing .ifc files to Visualize universal database and subsequent rendering.
ODAViewer	A sample application for iOS that renders database content and provides extended functionality compared to iVisualizeViewer such as import parameters, ability to save a file, etc. Written in QML and C++.
Nw2Visualize	Sample application (.tx module) that demonstrates importing Nw files (.nwc, .nwd, .nwf) formats to Visualize universal database and subsequent rendering.

Ps2Visualize	Sample application (.tx module) that demonstrates importing Ps files to Visualize universal database and subsequent rendering.
Prc2Visualize	Sample application (.tx module) that demonstrates importing a .prc file to Visualize universal database and subsequent rendering.
Visualize2Pdf	Sample application that demonstrates exporting 3D geometry that is open in a Visualize application to PDF file format.
VisualizeDatabaseDumper	A sample console application that print database content of a .vsf file to a console.
OdVisualizeFirstApp	A basic sample application that renders predefined content or a content from a loaded .obj file.

Differences Between Visualize and Visualize inWEB

	Visualize SDK	**Visualize inWEB**
Visualization of the different kinds of the data (2D, 3D, Point Cloud)	+	+
Editing (Not native formats)	+	+
Optimized performance Link to the video 	+	+
MT support	Reading + Vectorization	–
Platform	Desktop, mobile	Web, mobile
Intermediate (own) format	Yes (VSFX)	VSFX
PDF export + publishing	+	+ (with CDE inWEB)

	High-level SDK	Visualize inWEB
Visual styles	+	+
Improved text antialiasing	+	+
Shadows Link to the video 	+	+
SSAO (screen space ambient occlusion)	+	+
SMAA (subpixel morphological antialiasing)	+	+
Highlight customization	+	+
Reflection plane Link to the video 	+	+
GPU and fast CPU selection	+	+

	High-level SDK	Visualize inWEB
Import from various formats	+	+ (with CDE inWEB)
Native model hierarchy	+	+
Access to native properties	+	+
Export to DWG, OBJ, Navis	+	+ (with CDE inWEB)
Collision detection	+	+
GPU and CPU cutting planes	+	+
Animation	+	+
Progressive mesh	+	+
Partial viewing	+	+
Measurements	+	+

Working with Other Formats

Visualize provides a set of classes that store parameters used during import and export processes.

There are a number of different supported file formats.

Format	Export	Import
.dgn	no	yes
.dwf, .dwfx (Drawing Web Format)	no	yes
.dwg, .dxf	yes (.dwg only)	yes
.dae (Collada)	no	yes*
.ifc	no	yes
.nwd, .nwc, .nwf (NwInterop)**	yes	yes
.obj	yes	yes
.pdf (2D)	yes	no
.pdf (3D PRC-based)	yes	no
.prc	no	yes
.rfa, .rvt (BIM)**	no	yes

.rcs (Point Cloud)	no	yes
.sat (ACIS, Version 7 only)	no	yes
.stl	no	yes
.u3d	no	yes
.vsf (Visualize Stream File)	yes	yes
.xml	yes	no

* Direct import to Visualize from DAE is not currently implemented. DAE can be visualized using the module for importing from .dae to .dwg.

** BIM and NwInterop are not part of a standard ODA membership and require an additional annual licensing fee.

Pricing

ODA operates under a membership-based model, where the membership fee is per company, not per developer. You can find all pricing details on our website.

Membership Levels:

- **Commercial:** Ideal for desktop/mobile applications with fewer than 100 copies of the end-user product distributed annually. Only the Core Package is available at this level.

- **Sustaining:** Suitable for unlimited distribution of desktop/mobile applications or web/SaaS solutions. This level allows access to various Extensions in addition to the Core Package.
- **Founding:** Offers access to the source code of the SDKs, along with all the benefits of the Sustaining membership.

Visualize SDK is part of the Core Package, making it available to all membership levels.

What is the Core Package?

The Core Package includes access to the Drawing, Architecture, IFC, STEP, and Publish SDKs. Please note that individual products from the Core Package cannot be purchased separately.

What are Extensions?

Extensions are additional products that can be licensed separately:

BimRv: supports .rvt/.rfa formats.

BimNv: supports .nwc, .nwd, .nwf formats.

Civil, Mechanical, Map: extensions for the respective .dwg formats.

Subscription to each of these Extensions requires an additional fee and is available only to Sustaining, Founding, or Corporate members.