Digital Preservation Home | Digital Formats Home

Sustainability of Digital Formats: Planning for Library of Congress Collections

Search this site
Go

Introduction | Sustainability Factors | Content Categories | Format Descriptions | Contact
Format Description Categories >> Browse Alphabetical List

Wavefront OBJ File Format

>> Back

Table of Contents

Format Description Properties

- ID: fdd000507
- Short name: OBJ
- Content categories: model
- Format Category: file-format, encoding
- Other facets: unitary, text, structured, symbolic
- Last significant FDD update: 2020-01-23
- Draft status: Full

Identification and description

Full name	Wavefront OBJ File Format
Description	The Wavefront OBJ format is a format for defining the 3D geometry for the surface of one or more objects. The format was first used by Wavefront Technologies around 1990 and a specification was published to encourage interoperability. The introduction to the specification stated, "Object files define the geometry and other properties for objects in Wavefront's Advanced Visualizer. Object files can also be used to transfer geometric data back and forth between the Advanced Visualizer and other applications." Since the mid 1990s, the ASCII-based format has been treated as a vendor-neutral format, referred to as 'Wavefront OBJ," "Alias/Wavefront OBJ," or simply "OBJ." In 2020, the format remains widely used, particularly for 3D printing of objects in multiple colors. Note: The original specification for the OBJ file employs an associated Wavefront Material Template Library (MTL) file to define both colors and textures. Textures and colors specified by name in an MTL file are invoked in the OBJ file. See 3D Model Appearance factor below. The OBJ file format supports definition of the geometry for object surfaces using polygonal meshes or using freeform curves and surfaces. From a logical perspective, the central concept in an OBJ model is the "element." The specification lists the following as elements: point (*p*), line (*l*), face (*f*), curve (*curv*), 2D curve (*curv2*), and surface (*surf*). Elements are constructed from vertices; the vertices must be listed first in the file, so that the elements can refer to them by number, using the order in which they occur in the file. Elements can optionally be organized into groups for convenience or to guide rendering applications.

An OBJ file has a simple physical structure, consisting of lines that start with keywords. Following each keyword, appropriate options and values are specified. Lines beginning with # are comments. Long lines can be broken up, using a backslash (\) character at the end of lines to be continued. According to the Wavefront OBJ File Format Summary in the Encyclopedia of Graphics File Formats, 2nd Edition, "The most commonly encountered OBJ files contain only polygonal faces. To describe a polygon, the file first describes each point [vertex] with the "v" keyword, then describes the face with the "f" keyword. The line of a face command contains the enumerations of the points in the face, as *1-based indices* into the list of points, in the order they occurred in the file." [Note: A 1-based index means that the count of vertices begins at 1, rather than 0.] The description of Alias/WaveFront Object (.obj) File Format provided by Donald H. House for a course in computer graphics is a useful introduction to the keywords used for polygonal meshes; these keywords are supported by most applications that claim to support the OBJ format. Not all 3D software applications can handle freeform surfaces, which are defined using mathematical equations. Of those that can, support may be limited to the non-uniform rational basis spline (NURBS) model, although the OBJ specification allows for a variety of models.

Here is a simple example of an OBJ file, for a cube, taken from the specification

```
v 0.000000 2.000000 2.000000
v 0.000000 0.000000 2.000000
v 2.000000 0.000000 2.000000
v 2.000000 2.000000 2.000000
v 0.000000 2.000000 0.000000
v 0.000000 0.000000 0.000000
v 2.000000 0.000000 0.000000
v 2.000000 2.000000 0.000000
f 1 2 3 4
f 8 7 6 5
f 4 3 7 8
f 5 1 4 8
f 5 6 2 1
f 2 6 7 3
```

Many more examples can be found at John Burkardt's page on the OBJ format, for example, for a Mini Cooper.

Vertices for a polygonal model in an OBJ file can have three types of vertex data. See Wikipedia entry for Vertex (computer graphics).

- **v**: Defines the position of the vertex in three dimensions (x,y,z). Three floating point numbers. Required.
- **vn**: Vertex normal, a directional vector associated with a vertex, used to facilitate smooth shading. Three floating point numbers. Optional.
- **vt**: Texture coordinates, also known as UV coordinates. Typically two floating point numbers (u, v). These coordinates are used during rendering to determine how to paint the three-dimensional surface with pixels from a 2D texture map, e.g. an image in a format such as PNG. Optional.

For information on the keywords and options associated with freeform surface elements, see the specification and Useful References below.

OBJ files that contain only face elements and the associated vertex data are widely supported by 3D software applications, for import and for export. Nevertheless, there can be interoperability problems associated with applying colors and textures using an associated MTL file. An OBJ file refers to the MTL file in an *mtllib* declaration and definitions for objects (polygonal or freeform) within the OBJ file can refer to a particular named material definition with a *usemtl* statement. Some applications simply do not support the MTL files. The original Advanced Visualizer application required the MTL file to be in the same directory as the OBJ file and any texture map files also to be in the same directory. Some applications accept paths to files in other directories and others export the component files in separate directories and users have to move them before they can be successfully employed. Another potential source of interoperability problems with MTL files is the use of keywords not defined in

	the original MTL specification for proprietary use or to support new approaches for 3D rendering. For more details, see MTL. A convention for representing colors in an OBJ model without using an MTL file is supported by several applications. This convention supports a technique, known as *per-vertex* coloring, in which a color is associated with each vertex and pixels on a mesh face are colored by interpolation based on distance from the face vertices. This approach is of particular value for certain applications. When 3D models are derived by scanning (e.g., for a pathology specimen) or photogrammetry (e.g. for a 3D cultural heritage artifact), it is feasible to generate a single file which includes accurate color information. Another situation where this approach is valuable is for models that use basic colors to identify parts of a design model, or convey other information using artificial colors, for example, for an anatomical model. Having a single file can simplify workflows and storage for situations where texture maps are not needed. For more detail on the convention and its support in applications, see Note on per-vertex coloring below.
Production phase	The OBJ format can be an initial-state format, created by a 3D scanning process. It often serves as a middle-state format, created as output from software that supports 3D design. An OBJ model will often be input to a 3D printing process or to more advanced 3D modeling software to include in a 3D scene or animation. Models are often shared for re-use in the OBJ format.
Relationship to other formats	
May have component	MTL, Wavefront Material Template Library (MTL) File Format. An associated MTL file can define colors and texture for applying to the surface of the objects whose surface geometry is defined in the OBJ file.

Local use

LC experience or existing holdings	The OBJ format has been used in a pilot workflow for 3D capture of artifacts from the Kislak Collection.
LC preference	The Library of Congress Recommended Formats Statement (RFS) includes OBJ as an acceptable format for Scanned 3D Objects (output from photogrammetry scanning.

Sustainability factors

Disclosure	The OBJ format was developed by Wavefront Technologies around 1990 and the format was openly documented before 1995 as Appendix B1 of a manual for version 3 of the Advanced Visualizer software sold by Wavefront. This appendix has been made available in several places on the Internet and the Wikipedia entry for Wavefront .obj file states, "The file format is open and has been adopted by other 3D graphics application vendors." The originators were indeed happy to share the specification and the format is indeed supported in almost all CAD software. However, from a legal standpoint, the specification is probably proprietary to Autodesk; see Notes on History and Useful References below for details on the chain of mergers and acquisitions. Note that a binary equivalent, with a .mod extension, is mentioned in the introduction to the specification but was not documented.
Documentation	A specification for the Wavefront OBJ format was published in Appendix B1 of a manual for The Advanced Visualizer in the early 1990s. See Format Specifications below for additional copies of the same appendix.
Adoption	The Wavefront OBJ format is widely used for 3D printing. See, for example, question 40 in 3D Printing Guide: Basics and Most Common 3D File Formats from All3DP; and Working in 3D: Designing and Saving for 3D Printing, a guide from Auraria Library. According to OBJ File Format – Simply Explained for CAD and 3D Printing from All3DP, OBJ has wide export and import support among CAD software and in recent years, has become "the de facto file format for multi-color 3D printing because the otherwise standard 3D printing format STL does not support color and texture information."

Most CAD/CAM and 3D design applications can import and export polygonal meshes in the OBJ format. The compilers of this resource have not determined which widely used 3D software applications can work with freeform surfaces. See the 3D Model Geometry factor below for more detail.

As of 2019, with 3D models in wide use for video games, online product information, 3D printing, and in archaeology, medicine, and other professional disciplines, the simple, openly documented formats developed over 20 years ago, such as STL, OBJ, and PLY are supported in new contexts. For example, recent versions of the Windows and Mac operating systems come with basic tools that can render 3D files in a small number of formats, including the OBJ format. Windows 10 includes Paint 3D, which can open files in PLY, STL, OBJ, 3MF, FBX, and GLB (the binary variant of glTF_2_0) formats. In April 2017, Microsoft announced support for 3D formats (3MF, FBX, OBJ, PLY, STL) in OneDrive for Business. The version of Preview shipped with versions of Mac OS since 10.11 (El Capitan) has also been able to render 3D models in a number of formats. For example, Preview in Mac OS 10.14 (Mojave) supports PLY, OBJ, STL, USD (Universal Scene Description), ABC (Alembic), and possibly more.

3D modeling applications and conversion tools that can import, export, or convert OBJ files include: MeshLab, a widely used open-source software system for manipulating 3D mesh models, maintained by the Visual Computing Lab of ISTI-CNR; Blender, another popular free and open-source 3D creation suite, supported by the Blender Foundation, an independent non-profit public benefit corporation, established in the Netherlands; Mathematica; Rhino; Aspose.3D for Java; Assimp (Open Asset Import Library); quick3D and HOOPS Exchange SDK. Products from Autodesk that support the OBJ format include: Alias; Maya (import); Maya (export); Maya LT; and 3DS Max.

Recently, GIS applications have begun to integrate 3D support. Models in the OBJ format can be used in Esri's City Engine and via some extensions to ArcGIS Pro: ArcGIS Data Interoperability and 3D Analyst. The Feature Manipulation Engine (FME), which is widely used with GIS products, offers support for importing or converting the OBJ format.

Examples of software tools that have been developed specifically to read and write files in the OBJ format include: PyWavefront in Python; OBJloader.js in javascript, part of threejs toolkit; and file-format-wavefront for .NET.

Cultural heritage institutions and archaeologists have used the Wavefront OBJ format when scanning 3D artifacts from their sites or collections. However, many museums also use 3D formats that offer more advanced features. For example, the British Museum offers a queen from the Lewis chessmen for download from SketchFab in STL and OBJ, but also in glTF. SketchFab is used by many museums for making 3D models available. Wavefront OBJ is supported for uploading models to SketchFab and is one of the preferred formats listed at Supported 3D File Formats. The UK Archaeology Data Service lists the Wavefront OBJ format in a table of "common" formats, and describes it as "suitable for preservation."

Digital Imaging and Remote Sensing Image Generation (DIRSIG) is a specialized system that can be used to test image system designs, to create test imagery for evaluating image exploitation algorithms and for creating data for training image analysts. DIRSIG software has been developed over the course of over two decades for use by NSF, NASA, DoD and Intelligence community contractors and by universities doing research for these organizations. In addition to a custom format, the software will read the Alias/Wavefront OBJ file format directly. Other geometry formats must be exported to the Alias OBJ format for importation into a DIRSIG scene. DIRSIG support is only for meshes with faces with three or four vertices. See Alias/Wavefront OBJ File Format |Feature Support in DIRSIG. Another specialized system that uses OBJ and MTL files is the Channel Annotation Package (CHAP), a research tool for "functional annotation of novel ion channel structures that provides information on the biophysical properties of the ion permeation pathway by utilising molecular dynamics simulations." See Contents of the OBJ and MTL Output Files.

	A recent development is the approval in early 2020 by Working Group 17 for the Digital Imaging and Communications in Medicine (DICOM) standard, of Supplement 208: DICOM Encapsulation of OBJ Models for 3D Manufacturing and Virtual Reality. See slides in Supplement 208 from 2019, and DICOM Supplement Overview. Faced with interoperability problems in getting some 3D software to import the OBJ and MTL files together and yield an appropriate rendering, some users may choose a workflow that avoids the OBJ format. For more on sources of interoperability problems, see MTL.
Licensing and patents	There was no formal statement by Wavefront Technologies or by any of its succeeding corporate entities that the OBJ format may be used without a license. However, the original publication included the informal statement, "The .obj appendix is for those who want to use the .obj format to translate geometric data from other software applications to Wavefront products. It also provides information for Advanced Visualizer users who want detailed information on the Wavefront .obj file format." Comments welcome.
Transparency	The OBJ format is in plain text, and thus easy to view with basic editing tools. Use of freeform curves and surfaces rather than polygonal facets will yield statements that may be hard to interpret without an understanding of the mathematics that enables efficient representation and rendering of free-form surface patches. See Useful References below.
Self-documentation	There is no support for embedding structured metadata in an OBJ file. Comments, indicated by lines with a leading # character, can be included anywhere in the file.
External dependencies	For full rendering with color and textures as intended, an OBJ file may need one or more associated Material Template Library (MTL) files and supplementary texture files.
Technical protection considerations	The OBJ format has no internal capabilities for encryption or other technical protection.

Quality and functionality factors

Other	
3D Model Geometry	The OBJ file format can encode surface geometry of a 3D model, using a polygonal mesh and/or freeform geometry using any of a number of techniques to define curved surface patches: basis matrix; Bezier; B-spline; Cardinal spline; or Taylor expansion. See section on *Freeform curve and surface types* at page 65 of the specification for more detail. See also Useful References below for some links to explanations of the underlying mathematics. The compilers of this resource have been unable to determine how widely the more complex freeform modeling options are in use in OBJ files and how widely the techniques are supported in CAD software. The Wikipedia entry for Wavefront .obj files states, "The most common elements are geometric vertices, texture coordinates, vertex normals and polygonal faces." Comments welcome. OBJ File Format – Simply Explained for CAD and 3D Printing from All3DP states that in general, the most common freeform surface technique in use is NURBS (Non-Uniform Rational B-Spline), a particular variant of B-spline that the OBJ format does support. Alias from Autodesk does appear to handle freeform surfaces. However, the help for 3DS Max, also from Autodesk only mentions the use of meshes on import. Blender appears to have partial support for curved surfaces defined by NURBS, but converts them to meshes on output. Another example of limited freeform curve support is in NURBS-Python (geomdl), a cross-platform, object-oriented B-Spline and NURBS library written in Python, which can output OBJ files.
3D Model Appearance	Appearance characteristics such as color and texture can be defined in companion Material Template Library (MTL) files. The MTL file is a "library" that can contain one or more named material definitions, each of which can specify color, texture, and reflection characteristics. An OBJ file refers to the MTL file in an *mtllib* declaration and definitions for objects (polygonal or

	freeform) within the OBJ file can refer to a particular named material definition with a *usemtl* statement. The original specification for OBJ had no support for per-vertex color information. However, a convention has emerged, as described in the Geometric vertex section in the Wikipedia entry for the OBJ file format. For more detail on the convention and its support in applications, see Note on per-vertex coloring below.
3D Model Scene	An OBJ file can define several objects, but not a scene incorporating lighting positions or pre-determined viewing positions.
3D Model Animation	The OBJ format does not support pre-determined animation sequences or skeleton structures (assemblies of bones and joints). A 3D viewer that can render models in the OBJ format will allow rotation and zooming.

File type signifiers and format identifiers

Tag	Value	Note
Filename extension	obj	Defined in specification. See also IANA registration of 2020.
Internet Media Type	model/obj	See IANA registration of 2020.
Pronom PUID	fmt/1210	See http://www.nationalarchives.gov.uk/pronom/fmt/1210.
Wikidata Title ID	Q2119595	See https://www.wikidata.org/wiki/Q2119595.

Notes

General	**3D Model Characteristics**: The quality and functionality factors used above are based on an analysis from 2008 in An Overview of 3D Data Content, File Formats and Viewers, a technical report (ISDA08-002) from the National Center for Supercomputing Applications (NCSA). According to this analysis, the key aspects of a 3D model are: geometry, appearance, scene, and animation. For brief descriptions of these factors, see the table for 3D model formats under Quality and functionality factors. The descriptions there are drawn from the 2008 report and from 2019 Most Common 3D File Formats (an article from All3DP), which includes the OBJ format and uses the same factors to distinguish the capabilities of formats. All 3D formats support some method for representing the geometry (the shape) of an object; support for other aspects is optional depending on the primary use of the format. Table 3 in the 2008 report, An Overview of 3D Data Content, File Formats and Viewers, compares nine popular 3D formats in use in 2008, including OBJ, based on these four aspects of 3D models. **Potential interoperability issues:** The Wikipedia entry for the Wavefront .obj file notes that "OBJ files, due to their list structure, are able to reference vertices, normals, etc. either by their absolute position (1 represents the first defined vertex, N representing the Nth defined vertex), or by their relative position (-1 represents the latest defined vertex). However, not all software supports the latter approach, and conversely some software inherently writes only the latter form (due to the convenience of appending elements without needing to recalculate vertex offsets, etc.), leading to occasional incompatibilities." Documentation for the Wavefront OBJ Importer by Okino indicates that before the introduction of the MTL format, an earlier method for applying a texture map to a surface object was via a *usemap* statement in the OBJ file. The Okino importer would create equivalent MTL-style definitions for any texture maps referenced in this way. Comments welcome. Another source of potential interoperability issues is in associated MTL files. Some extensions to the MTL format have been introduced by software vendors

	or open source projects based on developments in computer graphics techniques for rendering surfaces. For example, in 2015, Exocortex proposed Extending Wavefront MTL for Physically-Based Rendering through adding new statements, e.g., for defining roughness and clearcoat thickness. See MTL for more details. **Per-vertex coloring in OBJ format:** The per-vertex coloring technique, which is supported in many 3D modeling applications, associates a color (usually based on RGB) with each vertex and renders colors for the pixels on a mesh face by interpolation based on distance from the face vertices. See On the storage of Color in meshes and What is a vertex color?. The Geometric vertex section in the Wikipedia entry for the OBJ file format describes a convention for storing colors for vertices in an OBJ model, listing color values immediately following the vertex coordinates. Three colors, in RGB order, are expressed as values in the range 0 to 1. For example, “v 1220.094482 -572.500000 177.713943 0.984314 0.000060 0.000050” demonstrates a vertex with color information near true red. Note that this convention allows coloring (but not texturing) of a 3D model without the need for an associated MTL file. There are advantages and disadvantages of using per-vertex colors depending on context and intended use. In one context, A game of Tricks II – Vertex Color explains how vertex colors can improve performance in video games. For a discussion of the complex choice between using per-vertex colors and texture mapping in the context of a cultural heritage artifact, see the tutorial video MeshLab Basics: Color encoding. Having the entire model in a single file can simplify workflows and storage organization, but consideration must be given to the importance of geometric precision and color accuracy, and effects on file size and computational efficiency for rendering and manipulation in the intended use. One example from the medical field where per-vertex coloring can be advantageous is for 3D scanning of a pathology specimen; the scanning operation can capture surface color information. Per-vertex coloring can also be useful for applying discrete colors or identifiers via color information on an anatomical model or a model for an assemply of parts. Comments welcome. Support for this convention for per-vertex coloring in the OBJ file format is provided by several applications, including Meshlab (see video on MeshLab Basics: Color encoding), Unity ProBuilder, Sketchfab, the Open Asset Import Library (Assimp), and the threejs library. See How to display and use Vertex Color from OBJ model for an informative discussion and evidence of support in Assimp. Versions of Adobe Photoshop since 2018 have been found to recognize colors on importing OBJ files using this convention for per-vertex colors. The Preview app on Mac OS also recognizes and renders the colors appropriately. See Useful References below for links to resources about per-vertex coloring, also termed "color per vertex" (CPV). Note that the convention described here is not the only mechanism that has been implemented to encode per-vertex colors in an OBJ file, but it is the most widely supported. Comments welcome.
History	The OBJ format was developed by Wavefront Technologies for the Visualizer products that it developed in the late 1980s and early 1990s. The compilers of this resource have not been able to determine an exact date for the first use of the format. The company history reveals that in 1988, Wavefront released its first desktop product, known as Personal Visualizer, for the Silicon Graphics (SGI) workstation. In 1989, came the Data Visualizer, intended for industrial design applications. Since a Programmer's Project Guide for the Advanced Visualizer was published by Wavefront Technologies in 1990, The Advanced Visualizer (TAV) software came out soon after. The origin of the specification that is widely found is Appendix B of a manual for Advanced Visualizer 3.0; and includes an implication that the OBJ format had been used in the previous version, but not documented. This specification indicated that "The curve and surface extensions to the .obj file format were developed in conjunction with mental images GmbH. Berlin, Germany, as part of a joint development project to incorporate free-form surfaces into Wavefront’s Advanced Visualizer." By 1995, the Advanced Visualizer User's Guide, Version 4.2 was published. By 1998, a user guide for scientific visualization on workstations at Georgia Tech indicated that version 4.3 of the Advanced Visualizer was installed, soon to be replaced by the first version of the successor product, Maya, which also used the OBJ format. On February 7, 1995, Silicon Graphics had announced that it would purchase Wavefront Technologies and Alias Research. SGI merged the two companies to

	create Alias|Wavefront, later simply Alias. In 2004, SGI sold Alias to a private equity firm, Accel-KKR, with the support of Alias management. See Alias | About: History. In 2006, Alias was acquired by Autodesk. See Wikipedia entry for Accel-KKR and a news release from Autodesk.

Format specifications

- The following URLs link to copies of the specification for the OBJ format that was distributed by Wavefront Technologies as Appendix B1 of a manual for version 3.0 of their Advanced Visualizer software. No copy is more authoritative than the others. The original date for this document is around 1992. See Notes on History above.
  - Copy of Appendix B1 of Advanced Visualizer manual | from Wavefront Technologies around 1992 (http://www.cs.utah.edu/~boulos/cs3505/obj_spec.pdf). Text as distributed with version 3 of Wavefront's Advanced Visualizer software.
  - Object Files (.obj) | from Paul Bourke's dataformats site. (http://paulbourke.net/dataformats/obj/). Appears to be a plain-text version of Appendix B1 from the manual for Wavefront's Advanced Visualizer software.
  - Wavefront Object Files | from Martin Reddy's Graphics File Formats site (http://www.martinreddy.net/gfx/3d/OBJ.spec). Another plain-text version of Appendix B1 from the manual for Wavefront&apos;s Advanced Visualizer software.
  - Object Files (.obj) | from Andrew Willis, The William States Lee College of Engineering (https://coefs.uncc.edu/arwillis/programming-and-documentation/object-files-obj/). Another copy of Appendix B1 from the manual for Wavefront's Advanced Visualizer software

Useful references

URLs

- Wikipedia entry for Wavefront .obj file (https://en.wikipedia.org/wiki/Wavefront_.obj_file).
- Informal descriptions of the format
  - Wavefront OBJ File Format Summary | from Encyclopedia of Graphics File Formats, 2nd ed. 1996 (https://www.fileformat.info/format/wavefrontobj/egff.htm).
  - Alias/WaveFront Object (.obj) File Format (https://people.cs.clemson.edu/~dhouse/courses/405/docs/brief-obj-file-format.html). This useful description covers only the keywords associated with polygonal mesh geometry
  - OBJ File Format – Simply Explained for CAD and 3D Printing | from All3DP (https://all3dp.com/1/obj-file-format-3d-printing-cad/). Includes comparison with STL, FBX, and Collada formats.
  - File Format Developer Guide | OBJ (https://wiki.fileformat.com/3d/obj/).
  - Autodesk Alias Products | OBJ format (https://knowledge.autodesk.com/support/alias-products/learn-explore/caas/CloudHelp/cloudhelp/2019/ENU/Alias-ImportingExporting/files/GUID-A1CC091A-A9C8-45DB-A3EA-2DAFE7E1A59D-htm.html).
  - Alias/Wavefront OBJ File Format (http://www.dirsig.org/docs/new/obj.html). Outlines how to use the Alias/Wavefront OBJ file format to import attributed geometry into DIRSIG
- Most Common 3D Printer File Formats (2020) (https://all3dp.com/3d-printing-file-formats/). Compares formats for 3D printing: STL, OBJ, AMF, and 3MF. With detailed pros and cons,
- Resources with sample OBJ files
  - OBJ Files: A 3D Object Format | example files (http://people.math.sc.edu/Burkardt/data/obj/obj.html).
  - Sample files: OBJ | from LaserDesign (https://www.laserdesign.com/sample-files/type/obj/).
  - Free 3D Models in OBJ format | from Free3D (https://free3d.com/3d-models/obj). Shows that OBJ is widely used as a format in which to share models.
- Selected tools designed to support conversion or manipulation of 3D models and which can import or export OBJ files
  - Blender | File Formats (https://www.blender.org/features/pipeline/). A free open-source 3D creation suite
  - Wavefront OBJ | in Blender application (https://docs.blender.org/manual/en/dev/addons/import_export/io_scene_obj.html). Implies that OBJ files can be imported and exported. On export, NURBS surfaces are converted to polygonal meshes. As of early 2020, Blender does not recognize per-vertex colors on import of OBJ objects employing the convention (described in Notes above) for storing per-vertex colors immediately following vertex coordinates.
  - MeshLab | from Visual Computing Lab of ISTI-CNR (http://www.meshlab.net/).
  - Open Asset Import Library (assimp) (https://assimp-docs.readthedocs.io/en/latest/about/intoduction.html). A library to load and process geometric scenes from various data formats.
  - Aspose.3D for Java (https://products.aspose.com/3d/java). Describes itself as a standalone gameware and CAD API to manipulate 3D files. Can import Wavefront OBJ files.
  - HOOPS Exchange SDK (https://www.techsoft3d.com/products/hoops/exchange/). Describes itself as a CAD Data Translation Toolkit. Supports OBJ for import and export.
  - Okino: Wavefront OBJ Importer (https://www.okino.com/conv/imp_wave.htm).
  - NURBS-Python | Supported File Formats (https://nurbs-python.readthedocs.io/en/latest/file_formats.html).

    - Integrate Wavefront OBJ Using FME (Feature Manipulation Engine) | from Safe Software (https://www.safe.com/integrate/obj/).
    - vtkplotter | Python module for scientific visualization, analysis and animation of 3D objects and point clouds. (https://vtkplotter.embl.es/index.html). Can import meshes from VTK format, STL, Wavefront OBJ, 3DS, PCD (PointCloud), volumetric TIFF stacks, 2D images PNG, JPEG.
- Utilities designed specifically to read or write Wavefront OBJ files
    - PyWavefront | python program to read OBJ files (https://github.com/pywavefront/PyWavefront).
    - file-format-wavefront | on github (https://github.com/dwmkerr/file-format-wavefront). A .NET library to load data from Wavefront *.obj files and associated .mtl files
    - OBJ Loader (Javascript code) | from Three.js Javascript library (https://threejs.org/examples/jsm/loaders/OBJLoader.js). three.js, a 3D library that tries to make it as easy as possible to get 3D content on a webpage, recommends use of glTF but provides loaders for other popular formats, including OBJ.
- The use of freeform curves and surfaces in 3D software for generating, viewing, or rendering, depends on sophisticated mathematical techniques. Full implementation of the OBJ format can require use of techniques described in the resources below.
    - Wikipedia entry for Bézier curve (https://en.wikipedia.org/wiki/B%C3%A9zier_curve).
    - Wikipedia entry for Bézier surface (https://en.wikipedia.org/wiki/B%C3%A9zier_surface).
    - Bezier and B-spline Surfaces | by Susan Laflin (1999) (https://www.cs.bham.ac.uk/~slb/courses/Graphics/g95.html).
    - About Bezier Splines | from Help for Windows GDI+, an API for C/C++ programmers (https://docs.microsoft.com/en-us/windows/win32/gdiplus/-gdiplus-bezier-splines-about).
    - Wikipedia entry for B-spline (https://en.wikipedia.org/wiki/B-spline). The Non-Uniform Rational B-Spline (aka NURBS) is an important variant of B-spline for 3D Modeling.
    - Wikipedia entry for Non-Uniform Rational Basis Spline (NURBS) (https://en.wikipedia.org/wiki/Non-uniform_rational_B-spline). The OBJ format supports use of NURBS to model surface patches.
    - About Cardinal Splines | from Help for Windows GDI+, an API for C/C++ programmers (https://docs.microsoft.com/en-us/windows/win32/gdiplus/-gdiplus-cardinal-splines-about).
- The resources listed below relate to the history of the corporations that have owned or contributed to the OBJ format, starting with Wavefront Technologies, which merged with Alias, and ending with Autodesk.
    - Wikipedia entry for Wavefront Technologies (https://en.wikipedia.org/wiki/Wavefront_Technologies). The company that first defined the OBJ format. After a series of mergers and acquisitions, the associated software products and, presumably, the OBJ specification are owned by Autodesk in 2019.e transferred the success
    - Wikipedia entry for Alias Systems Corporation (https://en.wikipedia.org/wiki/Alias_Systems_Corporation). This company was formed in 1995 when Alias Research and Wavefront were both acquired by Silicon Graphics (SGI). In 2004, SGI sold Alias to Accel-KKR a private equity firm.merged
    - Wikipedia entry for The Advanced Visualizer (TAV) (https://en.wikipedia.org/wiki/The_Advanced_Visualizer). Software for which Wavefront Technologies developed the OBJ format.
    - About Alias | History (https://web.archive.org/web/20040630210601/http://www.aliaswavefront.com/eng/about/history/2004.shtml). Includes announcement of 2004 sale of Alias to Accel-KKR
    - Wikipedia entry for Accel-KKR (https://en.wikipedia.org/wiki/Accel-KKR). Alias was part of this private equity firm's portfolio until January 2006, when it was acquired by Autodesk.
    - Accel KKR | Portfolio: Accel (https://www.accel-kkr.com/portfolio/alias/). Alias was part of the Accel KKR portfolio of companies through 2005.
    - Autodesk Completes Acquisition of Alias (January 2006) (http://investors.autodesk.com/news-releases/news-release-details/autodesk-completes-acquisition-alias).
- The resources below illustrate the continued popularity of the OBJ format. Some identify pros and cons in comparison with other 3D model formats.
    - Most Common 3D File Formats | from All3DP (https://all3dp.com/3d-file-format-3d-files-3d-printer-3d-cad-vrml-stl-obj/#Top).
    - The Ultimate Guide to 3D Printing – 101 Questions Answered | from All3DP (https://all3dp.com/3d-printing-3d-printer-guide-101-questions/). See question 40. Highlights STL, OBJ, PLY, and 3MF as the most common formats for 3D printing.
    - Should you use the OBJ 3D file format? (https://www.threekit.com/blog/should-you-use-the-obj-3d-file-format).
    - Working in 3D: Designing and Saving for 3D Printing | from Auraria Library guidance (https://guides.auraria.edu/c.php?g=567257&p=3907431). Recommends use of STL or OBJ.
    - What is the difference between STL and OBJ files and which one should I use? (2018) | from 3D Printing StackExchange (https://3dprinting.stackexchange.com/questions/5603/what-is-the-difference-between-stl-and-obj-files-and-which-one-should-i-use).
    - File types supported for previewing files in OneDrive, SharePoint, and Teams | from Microsoft (https://support.office.com/en-us/article/file-types-supported-for-previewing-files-in-onedrive-sharepoint-and-teams-e054cd0f-8ef2-4ccb-937e-26e37419c5e4).
    - Announcing New File Viewers Available for OneDrive For Business (April 2017) (https://techcommunity.microsoft.com/t5/OneDrive-for-Business/Announcing-New-File-Viewers-Available-for-OneDrive-For-Business/m-p/60040). Added support for 3D formats (3MF, FBX, OBJ, PLY, STL).

    - Channel Annotation Package (CHAP) | Contents of the OBJ and MTL Output Files (https://www.channotation.org/docs/contents_objmtl_file/). OBJ/MTL are used in CHAP to represent the geometry of a channel's permeation pathway and its colour coding according to various properties such as channel radius or channel hydrophobicity.
- The resources below relate to support for OBJ in various Autodesk products
    - 3DS Max | Wavefront (OBJ) Files (https://knowledge.autodesk.com/support/3ds-max/learn-explore/caas/CloudHelp/cloudhelp/2020/ENU/3DSMax-Data-Exchange/files/GUID-87B0A138-8348-4DCB-87EE-585841A27318-htm.html).
    - 3DS Max | OBJ Import Options Dialog (https://knowledge.autodesk.com/support/3ds-max/learn-explore/caas/CloudHelp/cloudhelp/2020/ENU/3DSMax-Data-Exchange/files/GUID-7A9535E6-235D-459B-B32D-5A9E5E4EBF28-htm.html). States that the OBJ/MTL export/import plug-in is provided by software from GuruWare.
    - 3DS Max | Wavefront Object (OBJ) Export Options Dialog (https://knowledge.autodesk.com/support/3ds-max/learn-explore/caas/CloudHelp/cloudhelp/2020/ENU/3DSMax-Data-Exchange/files/GUID-437051A1-983B-4B6A-80C4-D2FA43B7393D-htm.html).
    - Alias | OBJ format (https://knowledge.autodesk.com/support/alias-products/learn-explore/caas/CloudHelp/cloudhelp/2019/ENU/Alias-ImportingExporting/files/GUID-A1CC091A-A9C8-45DB-A3EA-2DAFE7E1A59D-htm.html). Suggests that Alias can import and export models with freeform surfaces, e.g., those defined by NURBS models.
- OBJ models can be imported into some products from ESRI, that are beginning to integrate 3D models with geospatial data.
    - CityEngine | Import OBJ (Wavefront) (https://doc.arcgis.com/en/cityengine/latest/help/help-import-obj.htm).
    - CityEngine | Export OBJ (Wavefront) (https://doc.arcgis.com/en/cityengine/latest/help/help-export-obj.htm).
    - ArcGIS Data Interoperability (https://www.esriuk.com/en-gb/arcgis/products/arcgis-data-interoperability/supported-formats). With this ArcGIS Pro extension, Wavefront OBJ as supported by "direct read" and for import and export.
    - ArcGIS 3D Analyst Toolbox | Import 3D Files (https://pro.arcgis.com/en/pro-app/tool-reference/3d-analyst/import-3d-files.htm). This extension to ArcGIS PRO includes support for importing Wavefront OBJ files into a 3D Scene.
    - How To: Use 3D shapes as marker symbols in an ArcGIS Pro scene view (https://support.esri.com/en/technical-article/000016699). ArcGIS Pro supports COLLADA (.dae), 3ds Max (.3ds), OpenFlight (.flt), and Wavefront (.obj) models for custom 3D symbols.
- The resources listed below discuss the use of per-vertex coloring in contrast to other techniques for specification of colors for 3D models.
    - On the Storage of Color in Meshes (2009) | from Meshlab Stuff blog (http://meshlabstuff.blogspot.com/2009/03/on-storage-of-color-in-meshes.html). Describes different ways to store coloring specification for a mesh.
    - What is a vertex color? (2017) | Q&A from Game Development Stack Exchange (https://gamedev.stackexchange.com/questions/139059/what-is-a-vertex-color).
    - How Do I Shade / Texture Stuff?: Shading Workflow Overview -- Pattern Storage And Texture Baking | from Computer Graphics Educaton by Neil Blevins (http://www.neilblevins.com/cg_education/shading_pipeline_overview/shading_pipeline_overview_6.htm). Gives brief explanations, with advantages and disadvantages, of 7 different methods for applying colors/textures/patterns.
    - A game of Tricks II – Vertex Color | from alkemi-games.com (http://www.alkemi-games.com/a-game-of-tricks-ii-vertex-color/). Discusses performance advantage of per-vertex coloring in video games.
- The resources below relate to the convention extending the OBJ format to allow specification of coloring within an OBJ model on a per vertex basis. The convention is supported by several widely used 3D applications and software libraries. See Note on per-vertex coloring above.
    - Answer to: How can I include vertex color information in .OBJ files? | from Game Development Stack Exchange (https://gamedev.stackexchange.com/questions/21303/how-can-i-include-vertex-color-information-in-obj-files/66270#66270).
    - MeshLab Basics: Color encoding (https://www.youtube.com/watch?v=HKipe2mv-NA&list=PL8B1E816EAE236B4D). Video tutorial introducing color encoding in Meshlab, including per-vertex coloring.
    - How to display and use Vertex Color from OBJ model | from Blender Stack Exchange (https://blender.stackexchange.com/questions/90890/how-to-display-and-use-vertex-color-from-obj-model).
    - OBJ Vertex Coloring Parsing Problem | Issue related to threejs software library (https://github.com/mrdoob/three.js/issues/2083). Thread about adding support for per-vertex color in OBJ format to threejs.
    - Export: OBJ-specific options | Help for Unity ProBuilder (https://docs.unity3d.com/Packages/com.unity.probuilder@4.0/manual/Object_Export.html?q=obj%20export#Obj). Includes option to write vertex colors when exporting OBJ files.
    - Uploading 3D Models: Vertex Colors | from Sketchfab (https://help.sketchfab.com/hc/en-us/articles/210765623-Vertex-Colors). Indicates that Sketchfab can recognize vertex colors in OBJ files on import.
- Working Group 17 for the Digital Imaging and Communications in Medicine (DICOM) standard is responsible for expanding the standard as a wrapper for 3D content. Based on a survey of use of 3D formats in medicine, support for encapsulating STL was added in DICOM 2018b. Support for encapsulating OBJ was added in DICOM 2020a. X3D/VRML and 3MF are also widely used and the working group expects to add support for encapsulating 3D models in more of these formats.
    - DICOM Working Group 17 (https://www.dicomstandard.org/wgs/wg-17/). Expansion of DICOM to support encapsulation of additional formats is planned.
    - DICOM Standard Status (updated with every change) (https://www.dclunie.com/dicom-status/status.html).

  - Overview: Encapsulation of OBJ Models for 3D Manufacturing and Virtual Reality | slides (2019) (https://www.dicomstandard.org/News/current/docs/sups/sup208-slides.pdf). Prepared to accompany ballot to add encapsulation of OBJ and MTL formats to DICOM standard.
  - Supplement 208: DICOM Encapsulation of OBJ Models for 3D Manufacturing and Virtual Reality (https://dicomstandard.org/News/ftsup/docs/sups/sup208.pdf).
- PRONOM entry for fmt/1210 (http://www.nationalarchives.gov.uk/pronom/fmt/1210). Information in PRONOM on Wavefront OBJ File. PUID: fmt/1210.
- Wikidata entry for Q2119595 (https://www.wikidata.org/wiki/Q2119595). Information in WikiData about Wavefront OBJ format. WikiData Title ID: Q2119595

Last Updated: 09/11/2020