

# X3D Graphics for Web Authors

## Chapter 13

### Geometry Nodes, Part 4: Triangles and Quadrilaterals

*There is no "royal road" to geometry.*

Euclid, to King Ptolemy I

# Contents

Chapter Overview and Concepts

X3D Nodes and Examples

Additional Resources

Chapter Summary and Suggested Exercises

References

# Contents

Chapter Overview and Concepts

X3D Nodes and Examples

Additional Resources

Chapter Summary and Suggested Exercises

References

# Chapter Overview

# Overview: Triangles and Quadrilaterals

## Common fields

- *solid, ccw, colorPerVertex, normalPerVertex*

Review from  
Chapters 2,6

## Normal vectors, Normal node

## Ordered polygonal nodes

- TriangleSet, TriangleFanSet, TriangleStripSet, and QuadSet

## Indexed polygonal nodes

- IndexedTriangleSet, IndexedTriangleFanSet, IndexedTriangleStripSet, and IndexedQuadSet

Pay close attention to detail with these nodes!

# Concepts

# Many different geometry nodes

An excellent aspect of X3D is that there are many different ways to create geometry

- Chapter 2, Geometry Primitives
- Chapter 6, Points, Lines and Polygons
- Chapter 10, Geometry2D Nodes
- Chapter 13, Triangles and Quadrilaterals

These are all handled consistently inside a Shape node with corresponding Appearance

# Motivation

Triangle and Quadrilateral nodes do not need to be tessellated (turned into triangles) by X3D viewers because they are already in that form

Polygons are defined in ways that can be directly passed to the underlying graphics hardware

Triangle nodes are perhaps the most low-level or fundamental nodes since they can identically represent any other polygonal node in X3D, if vertex definitions and connectivity ordering are properly expressed



# Triangles

Triangles are the primary low-level geometry construct used by graphics software, hardware

- More complex shapes are reduced to triangles by the rendering software
- A triangle is always planar, allowing the material appearance to fill it

Sometimes quadrilaterals are used, but problem is that values might be non-coplanar due to roundoff (or authoring) error

- Which means that filling in material is ill defined, and not properly or repeatably renderable

# Single-sided polygons

Graphics engines always prefer simplicity in order to achieve maximum run-time performance

- Top 3 considerations for graphics hardware: performance, performance, performance!

Single-sided polygons take about half the time to draw than double-sided polygons

- So if authors can arrange geometry so that only one side is ever visible to user, can go single-sided
- Technical term: *backface culling*
- Efficiency is rationale for many X3D default values
- Example: default setting is *solid='true'*
- Debugging hint: set *solid='false'* to show both sides

# Common field: *solid*

In 3D graphics, all triangles have 2 sides

- Graphics term: backface culling only draws front sides

The *solid* field defines whether a geometry node has an inside or not, with a default value of true

- *solid*='true' means do not render (draw) the inside
- *solid*='false' means render both inside and outside

This approach reduces the number of polygons needing to be drawn, thus improving performance

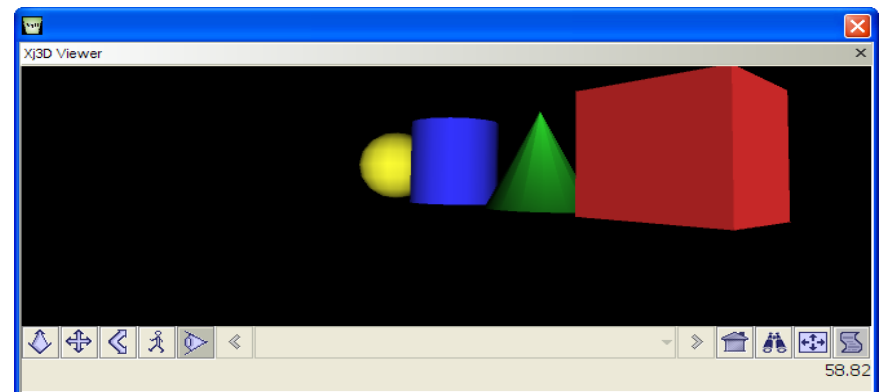
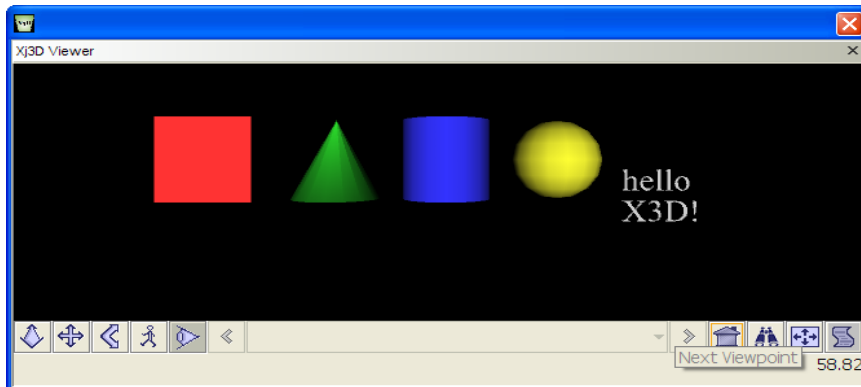
Confusing if user gets lost inside invisible geometry

- **Hint:** set *solid*='false' to draw both sides

## Common field: *solid*

To see an example of 'solid' geometry, rotate the GeometryPrimitives.x3d scene by 180 degrees

- Once rotated, the first four shapes remain visible, but the Text node disappears
- This is because *solid='true'* by default, so the reverse side of text is not drawn by default



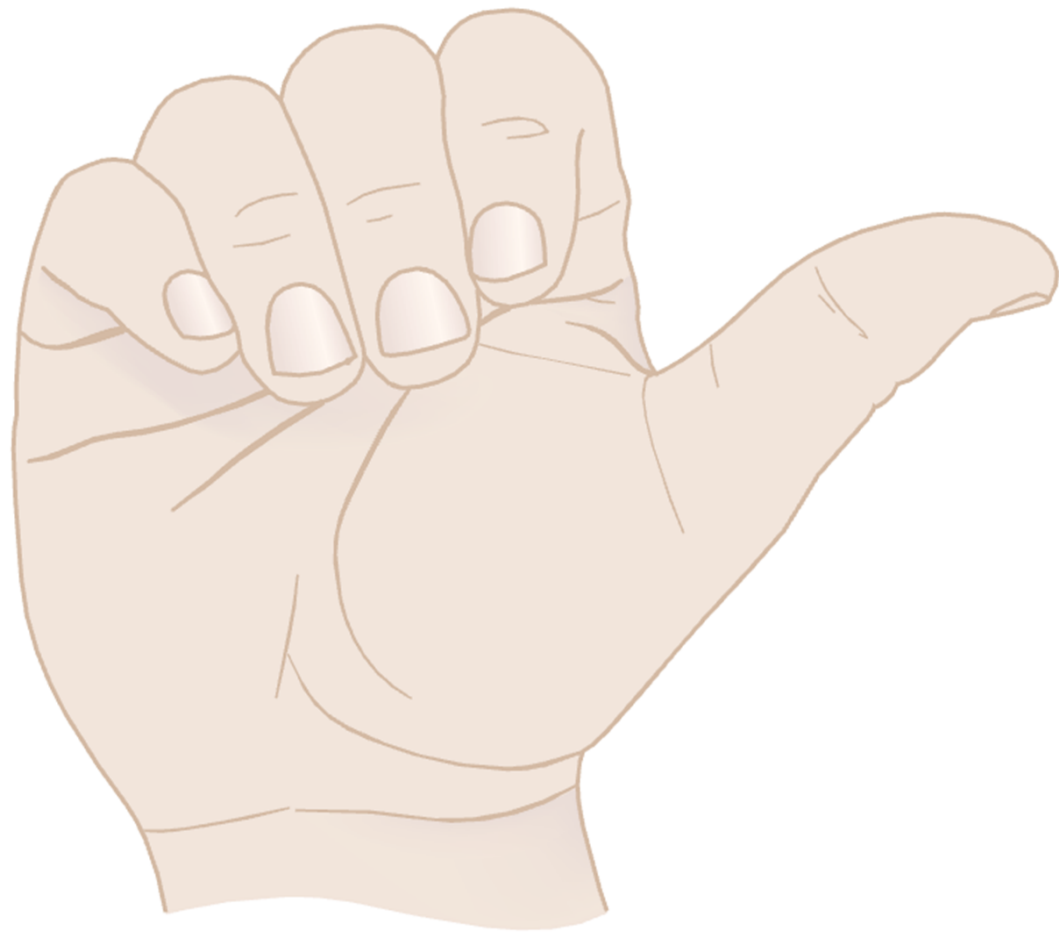
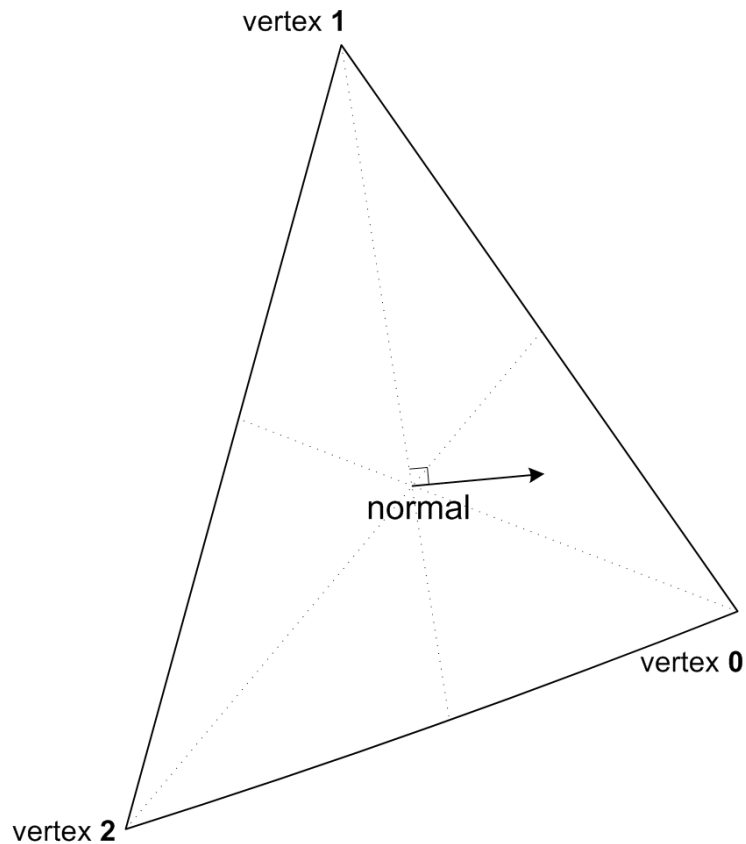
# Normal vectors

The *normal* vector is perpendicular to the face, pointing away from the centroid of polygon

Direction of normal vector defined by order of points defining the polygon and right-hand rule

- Align curvature of fingers to match polygon vertex points in order: indices 0, 1, 2 ...
- Thumb points in direction of (positive) normal, which is the front-facing side
- Negative normal thus points in direction of backface

# Right-hand rule for polygon normals



normal vector is at polygon centroid, with perpendicular direction according to right-hand rule

## Common field: *ccw*

*ccw* (counter clockwise) indicates whether default normal direction of polygons is counterclockwise (default) or clockwise

- *ccw*='true' is right-hand rule
- *ccw*='false' is opposite

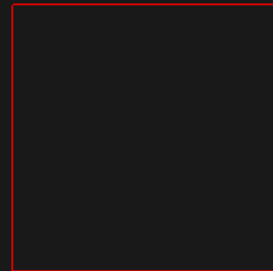
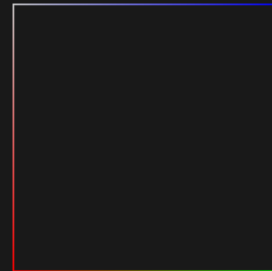
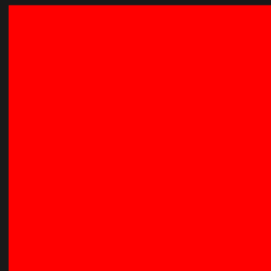
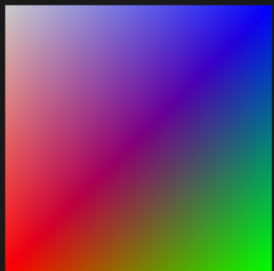
Hint: can correct some opposite-rendering geometry by reversing *ccw* value, rather than reordering all coordinates or indices

- Big time saver for some imports

## Common field: *colorPerVertex*

*colorPerVertex* indicates whether contained color values are applied to each vertex point (default), or to each polygonal face

- *colorPerVertex*='true' requires that # colors must equal # points
- *colorPerVertex*='false' requires that # colors must equal # polygons





# Review from Chapter 6

```
2 <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3 <X3D profile='Interchange' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/specifications/x3d-3.1.xsd'>
4 <head>
5 <meta content='ColorPerVertexExamples.x3d' name='title'//>
6 <meta content='Geometry Polygons Nodes: Color, Coordinate, ElevationGrid, Extrusion, IndexedFaceSet, IndexedLineSet, PointSet' name='description'//>
7 <meta content='Don Brutzman' name='creator'//>
8 <meta content='5 September 2005' name='created'//>
9 <meta content='5 September 2005' name='modified'//>
10 <meta content='Copyright (c) 2005, Daly Realism and Don Brutzman' name='rights'//>
11 <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ColorPerVertexExamples.x3d' name='identifier'//>
12 <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'//>
13 <meta content='../license.html' name='license'//>
14 </head>
15 <Scene>
16 <Viewpoint description='ColorPerVertex Examples' position='0 0 6'//>
17 <Background groundColor='0.1 0.1 0.1' skyColor='0.1 0.1 0.1'//>
18 <Transform translation='-0.5 0 0'>
19 <Transform translation='-3 0 0'>
20 <Shape>
21 <IndexedFaceSet colorIndex='0 1 2 3 0 -1' colorPerVertex='true' coordIndex='0 1 2 3 0 -1' solid='true'>
22 <Coordinate DEF='FourPoints' point='0 0 0 1 0 0 1 1 0 0 1 0'//>
23 <Color DEF='FourColors' color='1 0 0 0 1 0 0 0 1 0.8 0.8 0.8'//>
24 </IndexedFaceSet>
25 </Shape>
26 </Transform>
27 <Transform translation='-1 0 0'>
28 <Shape>
29 <IndexedFaceSet colorIndex='0' colorPerVertex='false' coordIndex='0 1 2 3 0 -1' solid='true'>
30 <Coordinate USE='FourPoints'//>
31 <Color USE='FourColors'//>
32 </IndexedFaceSet>
33 </Shape>
34 </Transform>
35 <Transform translation='1 0 0'>
36 <Shape>
37 <IndexedLineSet colorIndex='0 1 2 3 0 -1' colorPerVertex='true' coordIndex='0 1 2 3 0 -1'>
38 <Coordinate USE='FourPoints'//>
39 <Color USE='FourColors'//>
40 </IndexedLineSet>
41 </Shape>
42 </Transform>
43 <Transform translation='3 0 0'>
44 <Shape>
45 <IndexedLineSet colorIndex='0' colorPerVertex='false' coordIndex='0 1 2 3 0 -1'>
46 <Coordinate USE='FourPoints'//>
47 <Color USE='FourColors'//>
48 </IndexedLineSet>
49 </Shape>
50 </Transform>
51 </Transform>
52 </Scene>
53 </X3D>
```

**Edit Color**

containerField: color

DEF:  FourColors

USE:

color array

	r	g	b	color
0	1	0	0	
1	0	1	0	
2	0	0	1	
3	0.8	0.8	0.8	

+ - ↑ ↓

OK Cancel Help

## Common field: *normalPerVertex*

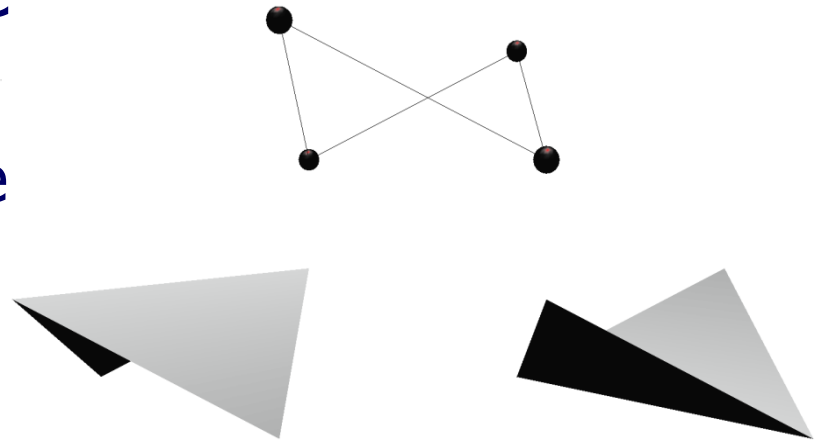
*normalPerVertex* indicates whether contained normal values are applied to each vertex point (default), or to each polygonal face

- *normalPerVertex*='true' requires that # normals must equal # points
- *normalPerVertex*='false' requires that # normals must equal # polygons

# Nonplanar quadrilaterals

Nonplanar quadrilaterals are ambiguously defined

- Can be triangulated in more than one way
- Can produce unpredictable results, rather than a consistent model for users



Nonplanar vertices in a single quadrilateral are thus undesirable and need to be avoided

# Common characteristics for these nodes

## Triangle and quad (quadrilateral) nodes

- Vertex values are defined in child `Coordinate` or `CoordinateDouble` node
- Colors are defined in child `Color` or `ColorRGBA` node
- May also have child `Normal` and `TextureCoordinate` (or `TextureCoordinateGenerator`) nodes

## Common attributes

- *ccw*, *solid*, *colorPerVertex*, *normalPerVertex*
- Indexed nodes include *index* integer array
- Animation possible using *set\_index* field
- **Caution:** some nodes use -1 sentinel *index* value but many don't, so always check indexing rules closely

# Distinct characteristics for each node

Color values follow the same indexing scheme as corresponding Coordinate point values

- TriangleSet, QuadSet: ordered sequence of values
- TriangleFanSet: *fanCount*
- TriangleStripSet: *stripCount*
- IndexedTriangleSet, IndexedQuadSet, IndexedTriangleFanSet, IndexedTriangleStripSet: all use common *index* array

Identical indexing correspondences are used for applying contained Normal, TextureCoordinate and TextureCoordinateGenerator node values

- No *colorIndex*, *normalIndex*, or *texCoordIndex* fields

# X3D Nodes and Examples

# Normal node

Normals are perpendicular vectors that are used in X3D lighting equations to shade geometry

Normals are typically computed automatically

- Since they can be unambiguously calculated from polygon coordinates
- Reduces file size and bandwidth requirements

Authors (or at least authoring tools) can precompute normal values and include them

- Possible reduction in load time
- Can enable application of special effects

# Effects of normal vectors

Recall that polygon lighting is computed by:

- projecting light rays from each light source, then
- reflecting them off intermediate polygons, until
- colored, modified light reaches user view

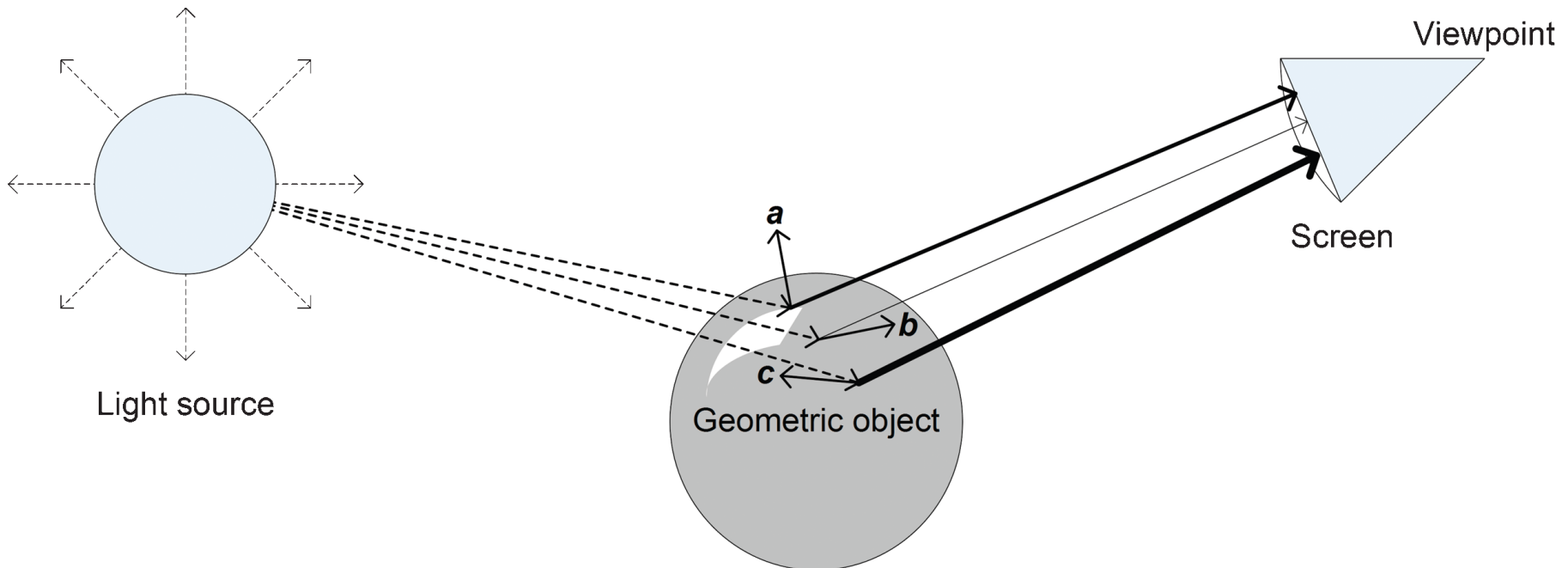
Incident and reflected light angles over the surface plane are also equal about the perpendicular

- Hence normal vector determines angle of reflection
- Direct straight-line reflections have greatest intensity
- Varying the direction of normal vectors can change the perceived brightness of each reflection, thus providing a special effect



# Normal vectors and lighting computations

Varying normal vector varies reflection intensity  
since brightest reflection is from smallest angle



Normal vector **a**: direction perpendicular to geometry, regular shading for result

Normal vector **b**: oblique direction away from light, lower intensity for result

Normal vector **c**: near-parallel direction towards light, higher intensity for result

# Unit normalization of normal vectors

Normalization is the process of changing the magnitude of a vector to have unit length

- Not same “normal” as perpendicular normal vector

Normal vector values should be unit normalized

- Allowing faster processing by graphics hardware
- Zero-magnitude vectors are degenerate, erroneous

Procedure:

- divide each vector component by magnitude

$$\text{Magnitude } r = \sqrt{(x^2 + y^2 + z^2)} \quad x' = x/r; y' = y/r; z' = z/r;$$

$$\text{Normalized vector } N' = (x', y', z')$$

# Normal-node hints and warnings

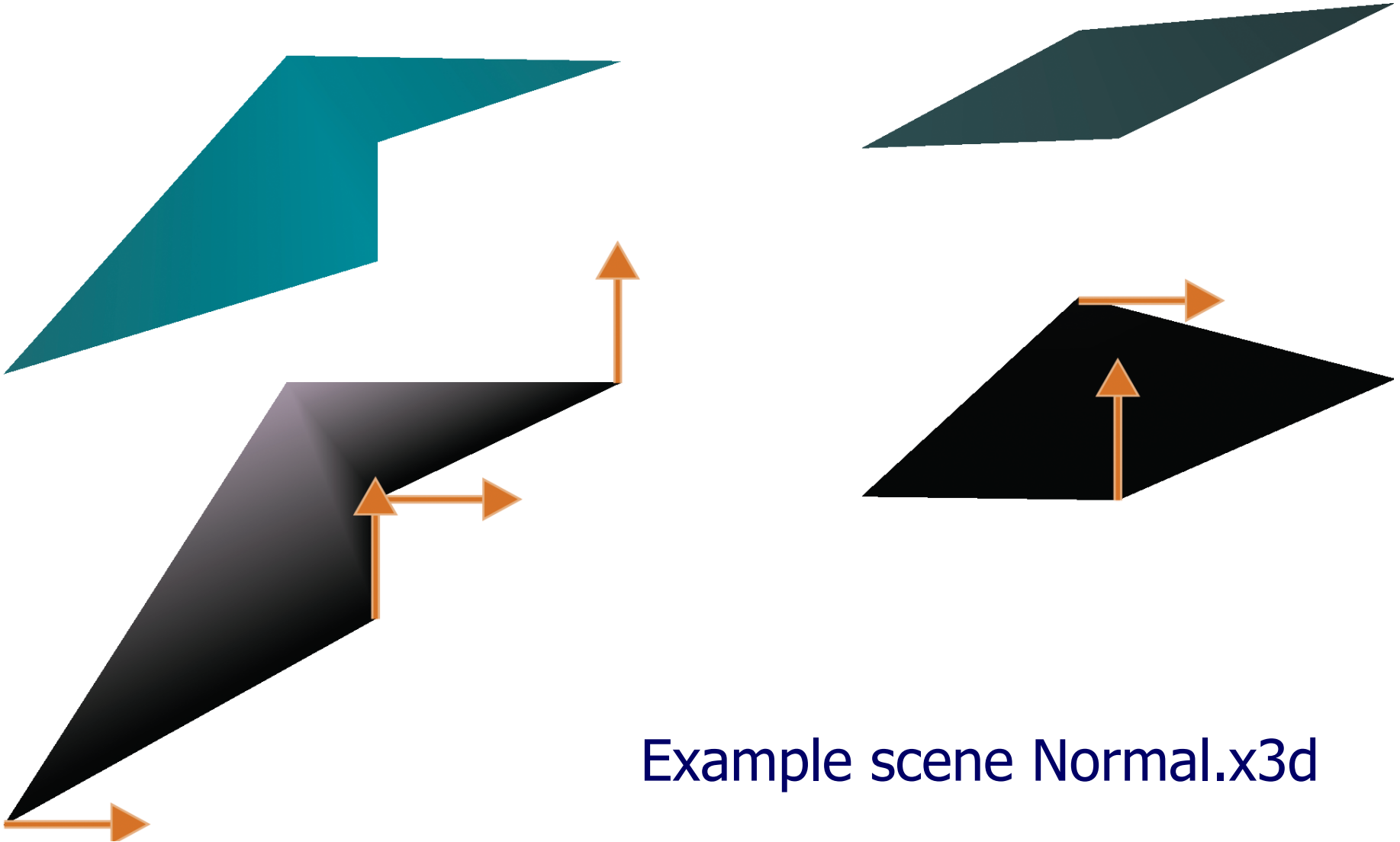
Adding normal values can reduce load time through reduced computation, but

- Computation is already quite fast
- File loading may take longer due to larger file size
- Network delay may be longer due to larger file size

Precomputing and animating normals may aid animations if

- Special rendering effects are desired
- Recomputation of normals becomes necessary due to changes in complex mesh geometry

# Example modification of normals



Example scene Normal.x3d



<input checked="" type="checkbox"/> <b>Normal</b>	<p><b>Normal</b> defines a set of 3D surface-normal vectors. Normal values are optional perpendicular directions, used per-polygon or per-vertex for lighting and shading.  <b>Hint:</b> used by <b>IndexedFaceSet</b> and <b>ElevationGrid</b>.</p>
DEF	<p><b>[DEF ID #IMPLIED]</b>  DEF defines a unique ID name for this node, referencable by other nodes.  <b>Hint:</b> descriptive DEF names improve clarity and help document a model.</p>
USE	<p><b>[USE IDREF #IMPLIED]</b>  USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children.  <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance.  <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
vector	<p><b>[vector: accessType inputOutput, type MFVec3f CDATA #IMPLIED]</b>  set of unit-length normal vectors, corresponding to indexed polygons or vertices.</p>
containerField	<p><b>[containerField: NMTOKEN "normal"]</b>  containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p><b>[class CDATA #IMPLIED]</b>  class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

# TriangleSet node

Most basic node, representing a set of triangles

- Vertex values are defined in child Coordinate or CoordinateDouble node
- Colors are defined in child Color or ColorRGBA node
- See Common Characteristics

No counting or indexing, no -1 sentinel values

- Each a-b-c sequence of *point* triplets in Coordinate node defines vertices of an independent triangle
- Coincident vertex values must be repeated, and so efficiency is reduced for large Coordinate nodes

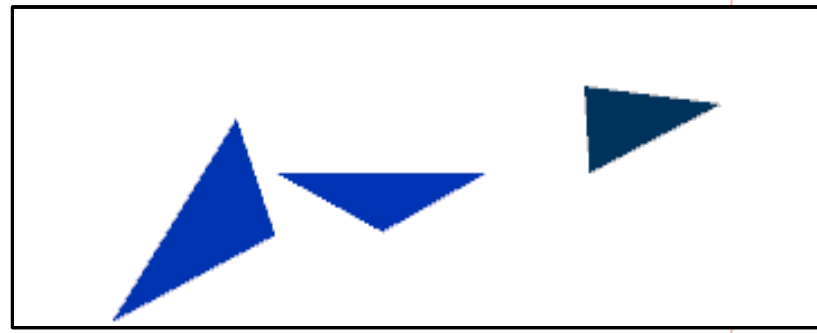


```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='ht
<head>
  <meta content='TriangleSet.x3d' name='title' />
  <meta content='A simple example of the use of the TriangleSet node.' name='description' />
  <meta content='22 May 2006' name='created' />
  <meta content='31 August 2008' name='modified' />
  <meta content='http://X3dGraphics.com' name='reference' />
  <meta content='http://www.web3d.org/x3d/content/examples/X3dResources.html' name='reference' />
  <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights' />
  <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject' />
  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/TriangleSet.x3d'
  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
  <meta content='../license.html' name='license' />
</head>
<Scene>
  <Viewpoint description='Book View 1' orientation='0 -1 0 0.05' position='0.13 2.51 11.24' />
  <Viewpoint description='Book View 2' orientation='-0.247 -0.964 -0.101 0.695' position='-4.96 3.13 7.09' />
  <Background skyColor='1 1 1' />
  <Transform>
    <Shape>
      <Appearance>
        <Material diffuseColor='0 0 1' emissiveColor='0 .2 0' shininess='.8' specularColor='
      <TriangleSet solid='false'>
        <Coordinate point='-4 1 3 -2 2 1 -3 4 0 0 2 0 2 3 1 -2 3 1 5 5 -2 4 3 1 6 4 2' />
      </TriangleSet>
    </Shape>
  </Transform>
</Scene>
</X3D>

```

Note no indexing, no -1 sentinel values



**Edit TriangleSet**

containerField

geometry

DEF

USE

ccw

colorPerVertex

normalPerVertex

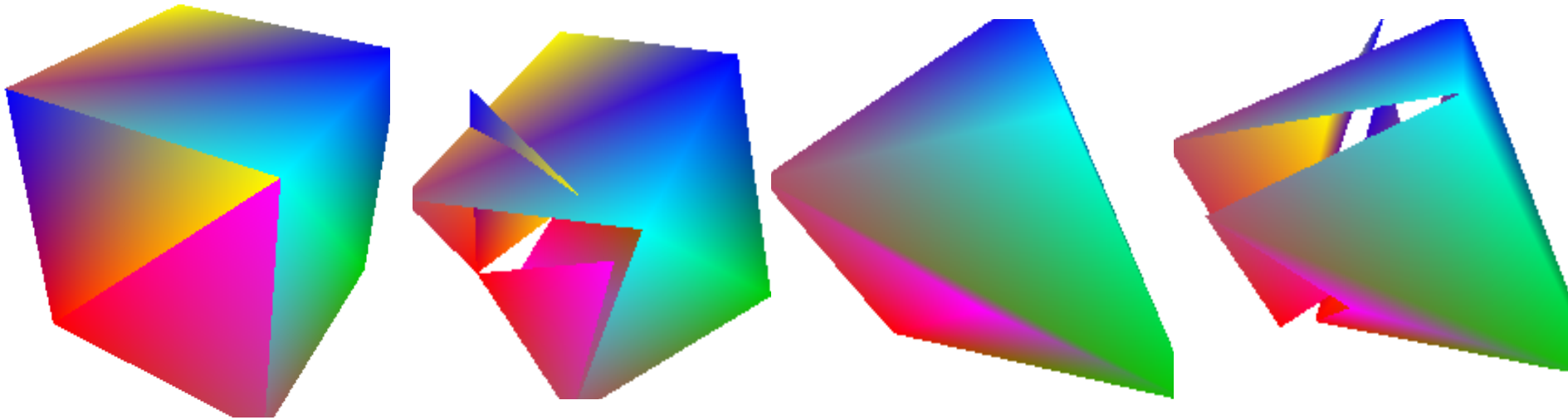
solid

OK Cancel Help





# MorphingTriangleSet example



Edit CoordinateInterpolator

containerField: children

DEF: Mover

USE:


Coordinate lists

36 coordinate(s) (column triples) Add coordinate columns Remove coordinate columns

key	0	1	2	3	4	5	6														
0	0	0	4	0	2	1.15	0	4	0	4	0	0	2	3.46	0	2	1.15	0	2	3.46	0
0.2	0	0	4	0	0	4	0	4	0	4	0	0	4	4	0	0	4	0	0	0	4
0.5	0	0	4	0	0	4	0	4	0	4	0	0	4	4	0	0	4	0	0	0	4
0.7	0	0	4	0	0	2	1.15	0	4	0	0	0	2	3.46	0	2	1.15	0	2	3.46	0

4 keyed coordinate list(s) (rows) Add keyed coordinate list Remove keyed coordinate list

OK Cancel Help

 <b>TriangleSet</b>	<b>TriangleSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.</b> <b>Hint:</b> insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
DEF	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. <b>Hint:</b> descriptive DEF names improve clarity and help document a model.
USE	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
ccw	<b>[ccw: accessType initializeOnly, type SBool (true false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.
colorPerVertex	<b>[colorPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
normalPerVertex	<b>[normalPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
solid	<b>[solid: accessType initializeOnly, type SBool (true false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
containerField	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

# TriangleFanSet node

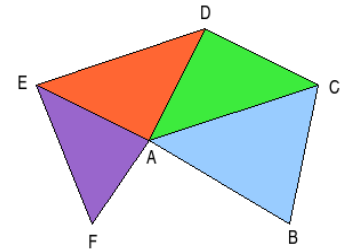
Efficient way to generate a group of triangles that share a single common vertex

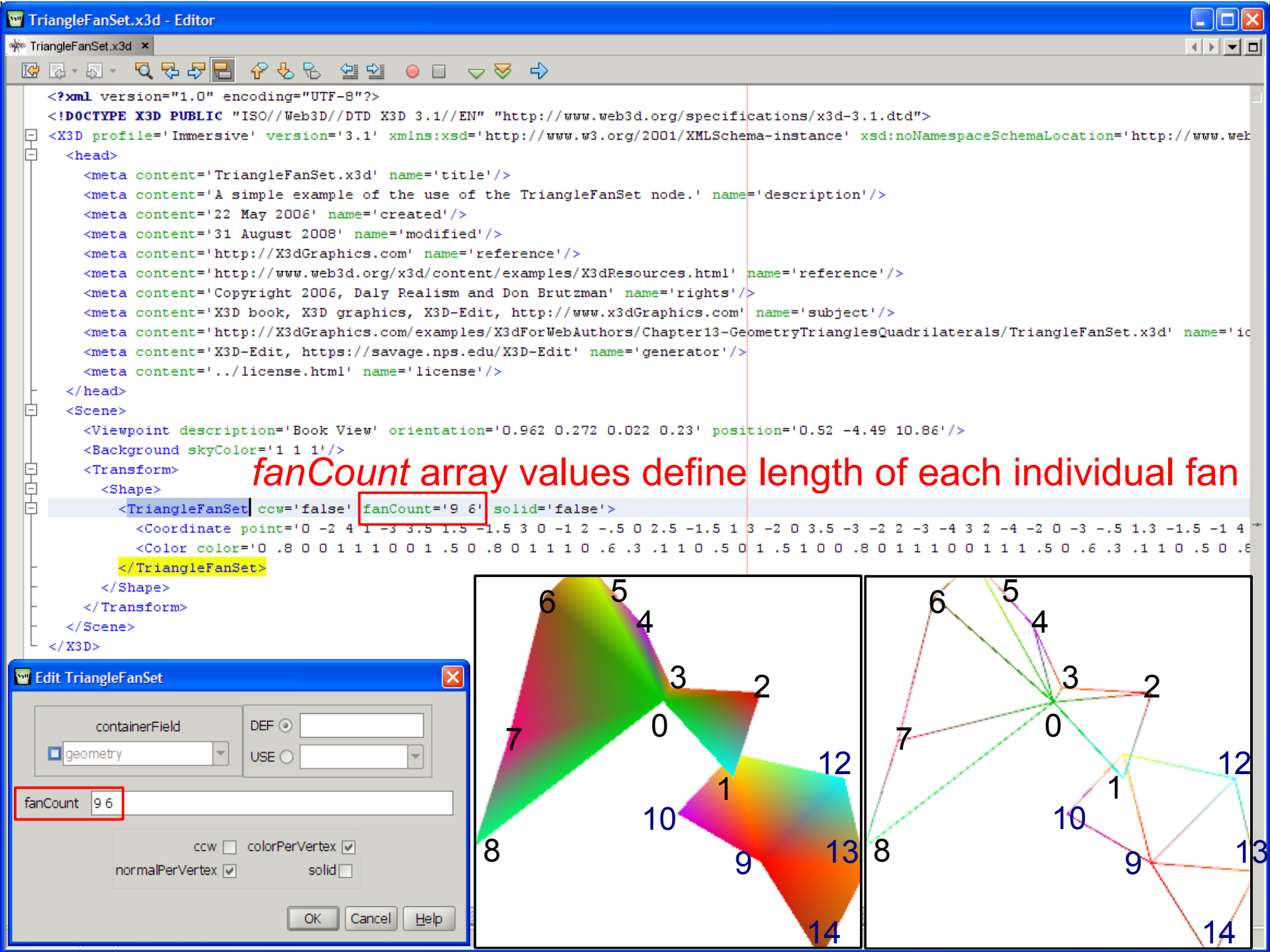
- Fan as in “hand-held fan”
- Each triangle created from latest point, preceding point, and initial center point



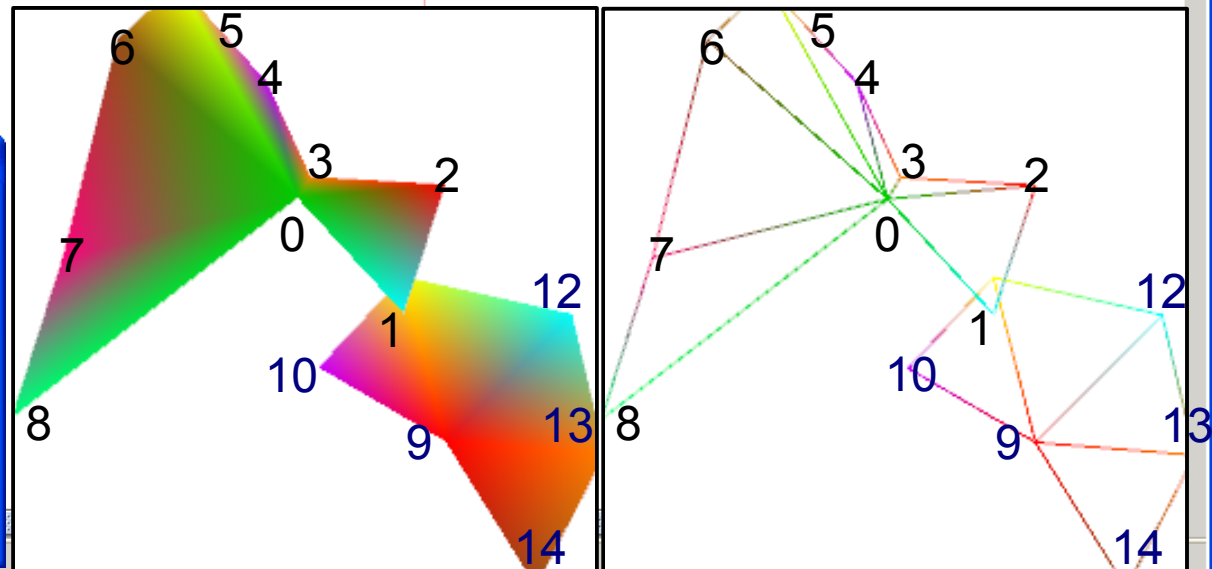
Indexing via *fanCount* array, no -1 sentinel values

- Each individual fan has  $(fanCount[i]-2)$  triangles
- Each  $a,b,c,\dots,n$  set of Coordinate *point* values define independent triangular fan
- Efficient for small pieces of geometry
- Coincident vertex values must be repeated, and so efficiency is reduced for large Coordinate nodes





*fanCount* array values define length of each individual fan




**Edit TriangleFanSet**

containerField DEF [ ]  
geometry [ ]  
USE [ ]

fanCount

ccw  colorPerVertex   
normalPerVertex  solid

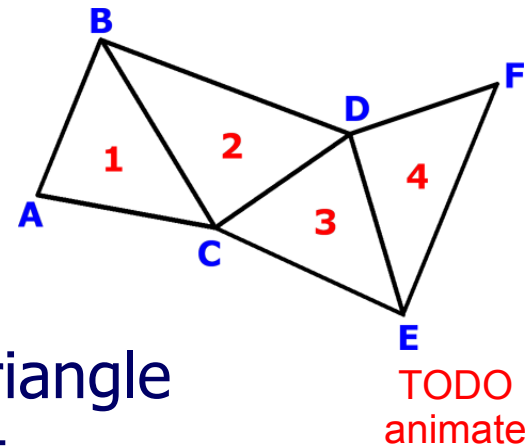
OK Cancel Help

 TriangleFanSet	<p>TriangleFanSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.  <b>Hint:</b> insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p><b>[DEF ID #IMPLIED]</b>  DEF defines a unique ID name for this node, referencable by other nodes.  <b>Hint:</b> descriptive DEF names improve clarity and help document a model.</p>
USE	<p><b>[USE IDREF #IMPLIED]</b>  USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children.  <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance.  <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
fanCount	<p><b>[fanCount: accessType initializeOnly, type MInt32 CDATA #IMPLIED]</b>  (3..+infinity) fanCount array provides number of vertices in each fan.</p>
ccw	<p><b>[ccw: accessType initializeOnly, type SBool (true false) "true"]</b>  ccw = counterclockwise: ordering of vertex coordinates orientation.  <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.</p>
colorPerVertex	<p><b>[colorPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b>  Whether Color node is applied per vertex (true) or per polygon (false).</p>
normalPerVertex	<p><b>[normalPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b>  Whether Normal node is applied per vertex (true) or per polygon (false).</p>
solid	<p><b>[solid: accessType initializeOnly, type SBool (true false) "true"]</b>  Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off).  <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!</p>
containerField	<p><b>[containerField: NMTOKEN "geometry"]</b>  containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p><b>[class CDATA #IMPLIED]</b>  class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

# TriangleStripSet node

Efficient way to generate geometry as a long strip of triangles

- Diagram of four triangles 1, 2, 3, 4, with vertices A, B, C, D, E, and F
- After first triangle, each subsequent triangle is defined by addition of a single point



Indexing via *stripCount* array, no -1 sentinel values

- Each individual strip has  $(stripCount[i]-2)$  triangles
- Each a,b,c,...,n sequence of Coordinate *point* values defines an independent triangular strip
- Coincident vertex values must be repeated, and so efficiency may be reduced for large Coordinate nodes



```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.wel
<head>
  <meta content='TriangleStripSet.x3d' name='title' />
  <meta content='A simple example of the use of the TriangleStripSet node.' name='description' />
  <meta content='22 May 2006' name='created' />
  <meta content='31 August 2008' name='modified' />
  <meta content='http://X3dGraphics.com' name='reference' />
  <meta content='http://www.web3d.org/x3d/content/examples/X3dResources.html' name='reference' />
  <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights' />
  <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject' />
  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/TriangleStripSet.x3d' name=
  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
  <meta content='../license.html' name='license' />
</head>
<Scene>
  <Viewpoint description='Initial' />
  <Viewpoint description='Book View' orientation='0 -1 0 0.05' position='0.13 2.51 11.24' />
  <Background skyColor='1 1 1' />
  <Transform>
    <Shape>
      <TriangleStripSet ccw='true' solid='false' stripCount='5 4'>
        <Coordinate point='-4 1 3 -2 2 1 -3 4 0 -2 3 1 0 4 0 2 3 1 5 5 -2 4 3 1 6 4 2' />
        <Color color='0 .8 0 0 1 1 1 0 0 1 .5 0 .8 0 1 1 1 0 .6 .3 .1 1 0 .5 0 1 .5' />
      </TriangleStripSet>
    </Shape>
  </Transform>
</Scene>
</X3D>

```

*stripCount* array values define length of each strip

Edit TriangleStripSet

containerField DEF [ ] USE [ ]

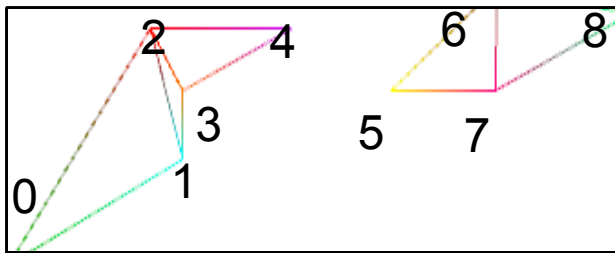
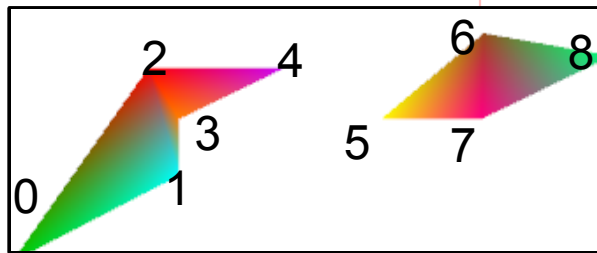
geometry

stripCount 5 4

ccw  colorPerVertex

normalPerVertex  solid

OK Cancel Help





 TriangleStripSet

	<p>TriangleStripSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node. <b>Hint:</b> insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p><b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. <b>Hint:</b> descriptive DEF names improve clarity and help document a model.</p>
USE	<p><b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children. <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
stripCount	<p><b>[stripCount: accessType initializeOnly, type MInt32 CDATA #IMPLIED]</b> (3..+infinity) stripCount array provides number of vertices in each strip.</p>
ccw	<p><b>[ccw: accessType initializeOnly, type SBool (true false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.</p>
colorPerVertex	<p><b>[colorPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).</p>
normalPerVertex	<p><b>[normalPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).</p>
solid	<p><b>[solid: accessType initializeOnly, type SBool (true false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!</p>
containerField	<p><b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p><b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

# QuadSet node

Defines a set of quadrilaterals (rectangles)

- Coordinate *point* value sequences need to be planar

No counting or indexing, no -1 sentinel values

- Each a-b-c-d set of *point* triplets in Coordinate node defines corners of an independent quad
- Coincident vertex values must be repeated, and so efficiency is reduced for large Coordinate nodes

Hint: must set CADGeometry component level 1



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http:
<head>
  <component level='1' name='CADGeometry' />
  <meta content='QuadSet.x3d' name='title' />
  <meta content='A simple example of the use of the QuadSet node.' name='description' />
  <meta content='31 August 2008' name='created' />
  <meta content='31 August 2008' name='modified' />
  <meta content='http://X3dGraphics.com' name='reference' />
  <meta content='http://www.web3d.org/x3d/content/examples/X3dResources.html' name='reference' />
  <meta content='Copyright 2008, Daly Realism and Don Brutzman' name='rights' />
  <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject' />
  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadri
  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
  <meta content='../license.html' name='license' />
</head>
<Scene>
  <Viewpoint description='QuadSet' />
  <Background skyColor='1 1 1' />
  <Transform>
    <Shape>
      <QuadSet solid='false'>
        <Coordinate point='-6 -2 0 -1 -2 0 -1 2 0 -6 2 0 1 -2 0 6 -2 0 6 2 0 1 2 0' />
        <Color color='0.2 0.2 0.2 1 0.058824 0.227451 0.4 1 0.121569 0 0.007843 0.960784 0.6 0.235294 0 0.94902 1 0.380392 0.23
      </QuadSet>
    </Shape>
  </Transform>
</Scene>
</X3D>
```

No indexing, no -1 sentinel values

**Edit Color**

containerField

color

DEF

USE

color array

r	g	b	...
0.2	0.2	0.2	
1.058824	0.227451		
0.4	1.121569		
0.007843	0.960784		
0.6	0.235294	0	
0.94902	1.0380392		
0.239216	1.0886275		
1.0121569	0.964706		

+ -

OK Cancel Help

**Edit QuadSet**

containerField

geometry

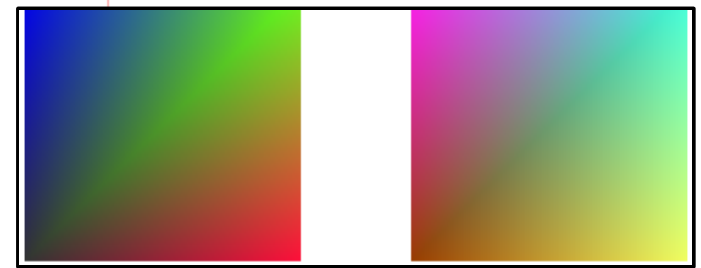
DEF

USE


ccw  colorPerVertex

normalPerVertex  solid

OK Cancel Help



3	2	7	6
0	1	4	5

 <b>QuadSet</b>	<b>[X3D 3.1] QuadSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.</b> <b>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</b>
<b>DEF</b>	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. <b>Hint:</b> descriptive DEF names improve clarity and help document a model.
<b>USE</b>	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children. <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
<b>ccw</b>	<b>[ccw: accessType initializeOnly, type SFBool (true false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.
<b>colorPerVertex</b>	<b>[colorPerVertex: accessType initializeOnly, type SFBool (true false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
<b>normalPerVertex</b>	<b>[normalPerVertex: accessType initializeOnly, type SFBool (true false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
<b>solid</b>	<b>[solid: accessType initializeOnly, type SFBool (true false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
<b>containerField</b>	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
<b>class</b>	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

# IndexedTriangleSet node


## Indexed version of TriangleSet node

- Triangles, triangles and nothing but more triangles!

Sequential triplets of *index* array values define triangles, so no -1 sentinel values are inserted

- Each sequential triplet from index array selects 3 Coordinate point values as independent triangle
- Thus no sentinel needed to distinguish triangles
- Efficient: each x-y-z point only needs to be defined once, since indexing can reuse it at any time



 <b>IndexedTriangleSet</b>	<b>IndexedTriangleSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.</b> <b>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</b>
DEF	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. <b>Hint:</b> descriptive DEF names improve clarity and help document a model.
USE	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children. <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
index	<b>[index: accessType initializeOnly, type MInt32 CDATA #IMPLIED]</b> (-1..+infinity) index specifies triangles by connecting Coordinate vertices.
ccw	<b>[ccw: accessType initializeOnly, type SBool (true false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.
colorPerVertex	<b>[colorPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
normalPerVertex	<b>[normalPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
solid	<b>[solid: accessType initializeOnly, type SBool (true false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
containerField	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

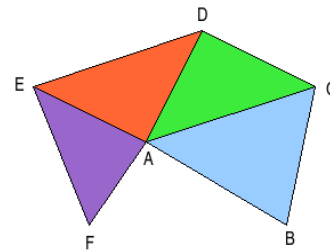
# IndexedTriangleFanSet node

Indexed version of TriangleFanSet node

- Just fans and more fans

*index* array values define each individual fan, separated by -1 sentinel values

- First value in each *index* subsequence is fan center
- Efficient: each x-y-z point only needs to be defined once, since indexing can reuse it at any time





```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.wel
<head>
  <meta content='IndexedTriangleFanSet.x3d' name='title'/>
  <meta content='A simple example of the use of the IndexedTriangleFanSet node.' name='description'/>
  <meta content='22 May 2006' name='created'/>
  <meta content='31 August 2008' name='modified'/>
  <meta content='http://X3dGraphics.com' name='reference'/>
  <meta content='http://www.web3d.org/x3d/content/examples/X3dResources.html' name='reference'/>
  <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights'/>
  <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject'/>
  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/IndexedTriangleFanSet.x3d'
  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
  <meta content='../license.html' name='license'/>
</head>
<Scene>
  <Viewpoint description='Book View' orientation='0 1 0 0.01' position='0.03 -1.84 10.04'/>
  <Background skyColor='1 1 1'/>
  <Transform>
    <Shape>
      <IndexedTriangleFanSet index='18 19 20 21 22 23 24 25 26 -1 27 28 29 30 31 32 -1' solid='false'>
        <Coordinate point='-4 1 3 -2 2 1.5 -3 4 0.5 -2 3 1.5 0 4 0 0 2 3 1.5 5 5 -2.5 4 3 1.5 6 4 2.0 -4 1 3 -2 2 1.0 -3 4 0.0 -2 3 1.0
        <Color color='0 .8 0 0 1 1 1 0 0 1 .5 0 .8 0 1 1 1 0 .6 .3 .1 1 0 .5 0 1 .5 1 0 0 1 .5 0 1 1 1 0 0 1 1 1 .5 0 .6 .3 .1 1 0 .5 0 .
      </IndexedTriangleFanSet>
    </Shape>
  </Transform>
</Scene>
</X3D>

```

*index selects fan coordinates, separated by -1 sentinel value*

Edit IndexedTriangleFanSet

containerField DEF [ ]

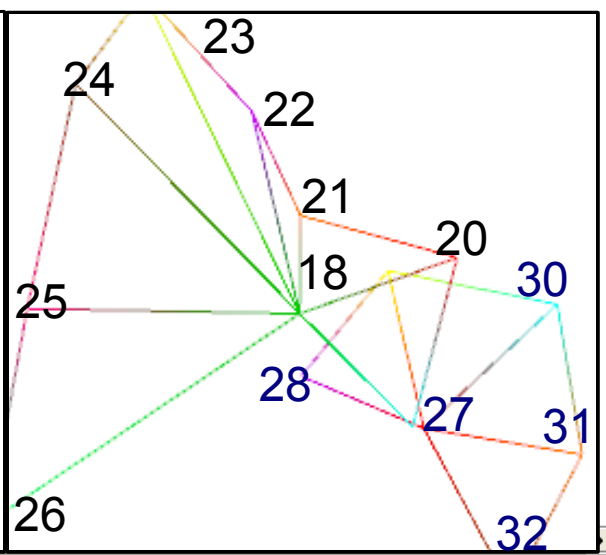
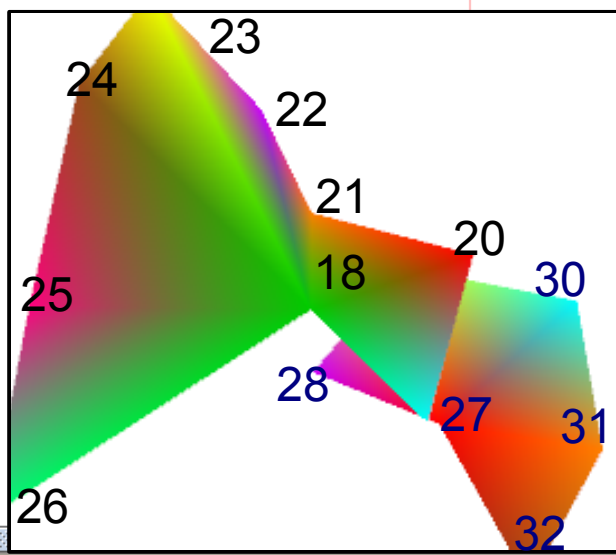
geometry USE [ ]


index 18 19 20 21 22 23 24 25 26 -1 27 28 29 30 31 32 -1

ccw  colorPerVertex

normalPerVertex  solid

OK Cancel Help

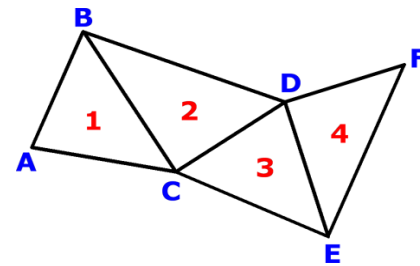


 IndexedTriangleFanSet	<p><b>IndexedTriangleFanSet</b> is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.  <b>Hint:</b> insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p><b>[DEF ID #IMPLIED]</b>  DEF defines a unique ID name for this node, referencable by other nodes.  <b>Hint:</b> descriptive DEF names improve clarity and help document a model.</p>
USE	<p><b>[USE IDREF #IMPLIED]</b>  USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children.  <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance.  <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
index	<p><b>[index: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</b>  (-1..+infinity) index specifies triangles by connecting Coordinate vertices.</p>
ccw	<p><b>[ccw: accessType initializeOnly, type SFBool (true false) "true"]</b>  ccw = counterclockwise: ordering of vertex coordinates orientation.  <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.</p>
colorPerVertex	<p><b>[colorPerVertex: accessType initializeOnly, type SFBool (true false) "true"]</b>  Whether Color node is applied per vertex (true) or per polygon (false).</p>
normalPerVertex	<p><b>[normalPerVertex: accessType initializeOnly, type SFBool (true false) "true"]</b>  Whether Normal node is applied per vertex (true) or per polygon (false).</p>
solid	<p><b>[solid: accessType initializeOnly, type SFBool (true false) "true"]</b>  Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off).  <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!</p>
containerField	<p><b>[containerField: NMTOKEN "geometry"]</b>  containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p><b>[class CDATA #IMPLIED]</b>  class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

# IndexedTriangleStripSet node

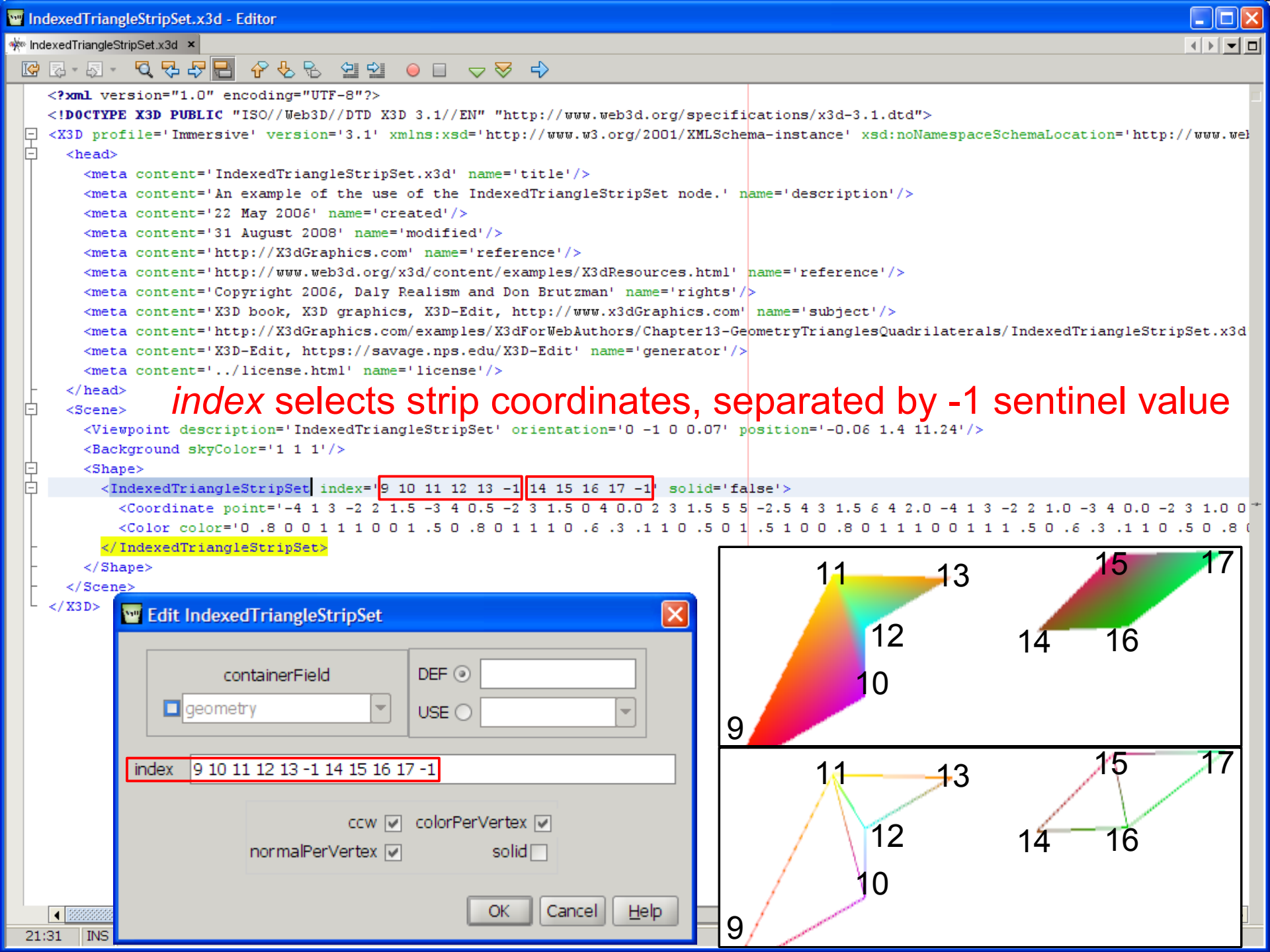
Indexed version of TriangleStripSet node

- Just a series of strip sets



*index* array values define each individual strip, separated by -1 sentinel values

- First three points referenced by *index* subsequence define initial triangle for strip
- Each subsequent index value adds another point, which in turn defines another triangle
- Efficient: each x-y-z point only needs to be defined once, since indexing can reuse it at any time



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.wel
<head>
  <meta content='IndexedTriangleStripSet.x3d' name='title' />
  <meta content='An example of the use of the IndexedTriangleStripSet node.' name='description' />
  <meta content='22 May 2006' name='created' />
  <meta content='31 August 2008' name='modified' />
  <meta content='http://X3dGraphics.com' name='reference' />
  <meta content='http://www.web3d.org/x3d/content/examples/X3dResources.html' name='reference' />
  <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights' />
  <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject' />
  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/IndexedTriangleStripSet.x3d'
  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
  <meta content='../license.html' name='license' />
</head>
```

*index selects strip coordinates, separated by -1 sentinel value*

```
<Scene>
  <Viewpoint description='IndexedTriangleStripSet' orientation='0 -1 0 0.07' position='-0.06 1.4 11.24' />
  <Background skyColor='1 1 1' />
  <Shape>
    <IndexedTriangleStripSet index='9 10 11 12 13 -1 14 15 16 17 -1' solid='false'>
      <Coordinate point='-4 1 3 -2 2 1.5 -3 4 0.5 -2 3 1.5 0 4 0.0 2 3 1.5 5 5 -2.5 4 3 1.5 6 4 2.0 -4 1 3 -2 2 1.0 -3 4 0.0 -2 3 1.0 0
      <Color color='0 .8 0 0 1 1 1 0 0 1 .5 0 .8 0 1 1 1 0 .6 .3 .1 1 0 .5 0 1 .5 1 0 0 .8 0 1 1 1 0 0 1 1 1 .5 0 .6 .3 .1 1 0 .5 0 .8
    </IndexedTriangleStripSet>
  </Shape>
</Scene>
</X3D>
```

**Edit IndexedTriangleStripSet**

containerField:  geometry

DEF:

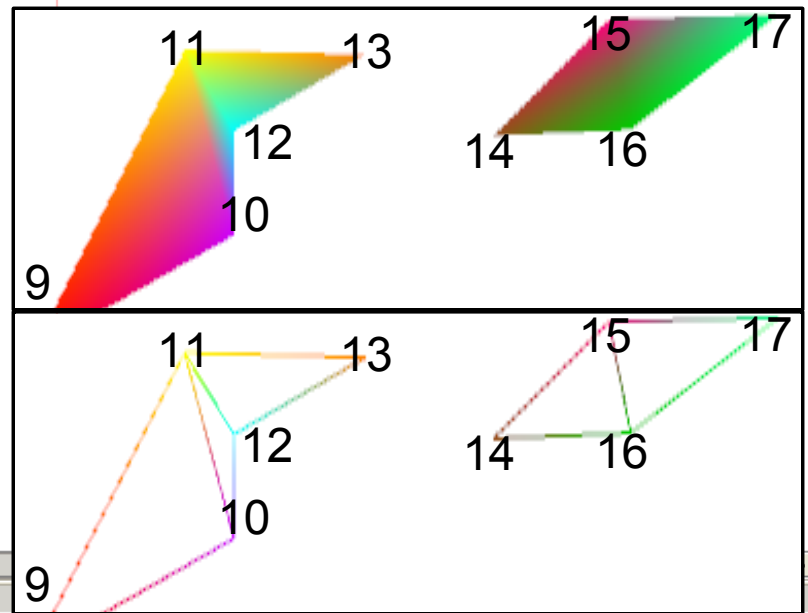
USE:


index:

ccw:  colorPerVertex:

normalPerVertex:  solid:

OK Cancel Help



 IndexedTriangleStripSet	<b>IndexedTriangleStripSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.</b> <b>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</b>
DEF	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. <b>Hint:</b> descriptive DEF names improve clarity and help document a model.
USE	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children. <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
index	<b>[index: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</b> (-1..+infinity) index specifies triangles by connecting Coordinate vertices.
ccw	<b>[ccw: accessType initializeOnly, type SFBool (true false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.
colorPerVertex	<b>[colorPerVertex: accessType initializeOnly, type SFBool (true false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
normalPerVertex	<b>[normalPerVertex: accessType initializeOnly, type SFBool (true false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
solid	<b>[solid: accessType initializeOnly, type SFBool (true false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
containerField	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

# IndexedQuadSet node

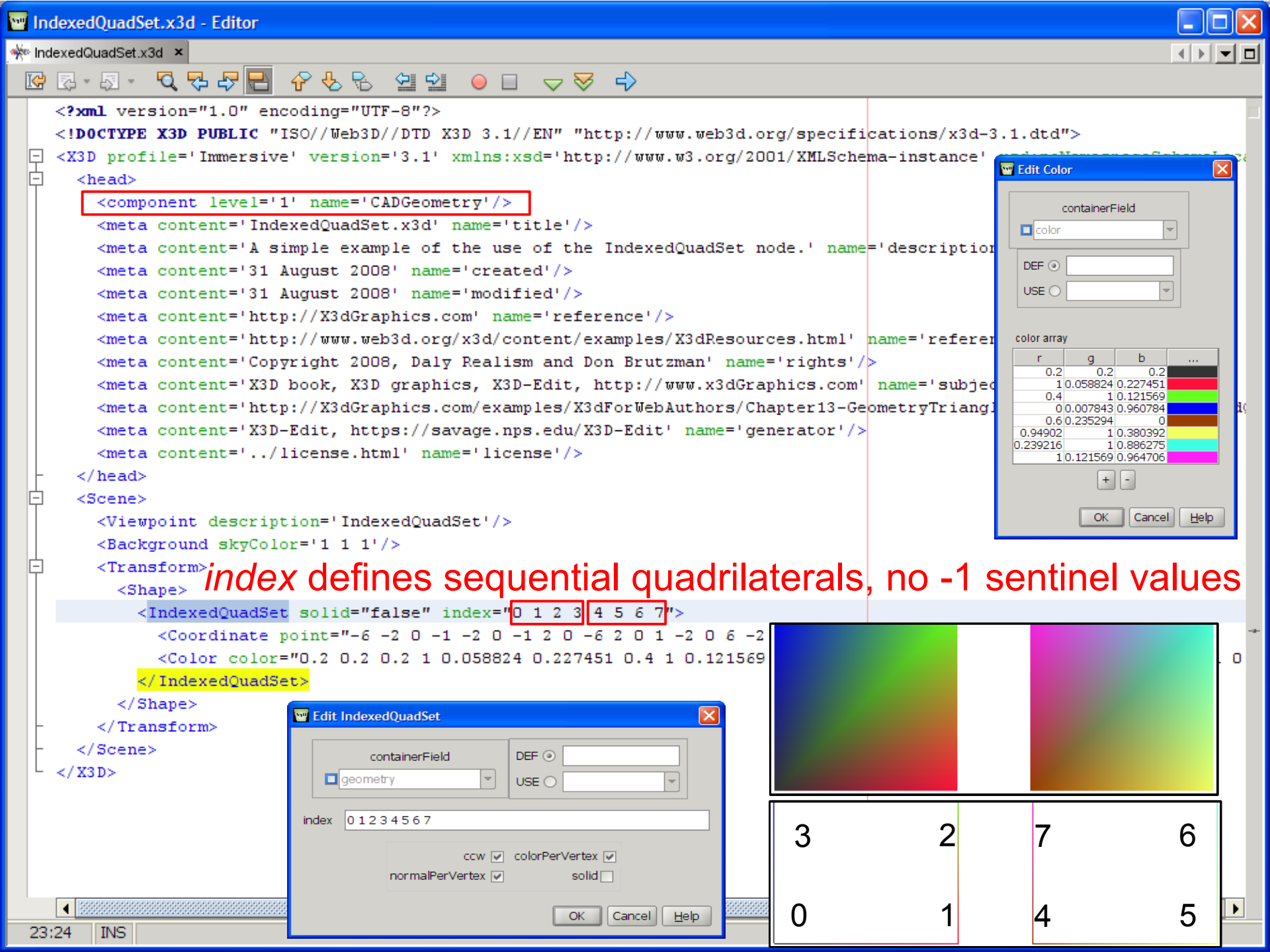
Defines a set of quadrilaterals (rectangles)

- Coordinate *point* value sequences need to be planar

Sequential 4-tuples of *index* array values define quads, so no -1 sentinel values are inserted

- Each a-b-c-d set of *point* quadruplets in Coordinate node defines corners of an independent quad
- Efficient: each x-y-z point only needs to be defined once, since indexing can reuse it at any time

Hint: must set CADGeometry component level 1



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance'>
  <head>
    <component level='1' name='CADGeometry' />
    <meta content='IndexedQuadSet.x3d' name='title' />
    <meta content='A simple example of the use of the IndexedQuadSet node.' name='description' />
    <meta content='31 August 2008' name='created' />
    <meta content='31 August 2008' name='modified' />
    <meta content='http://X3dGraphics.com' name='reference' />
    <meta content='http://www.web3d.org/x3d/content/examples/X3dResources.html' name='reference' />
    <meta content='Copyright 2008, Daly Realism and Don Brutzman' name='rights' />
    <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject' />
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTriang' name='subject' />
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
    <meta content='../license.html' name='license' />
  </head>
  <Scene>
    <Viewpoint description='IndexedQuadSet' />
    <Background skyColor='1 1 1' />
    <Transform>
      <Shape>
        <IndexedQuadSet solid="false" index="0 1 2 3 4 5 6 7">
          <Coordinate point="-6 -2 0 -1 -2 0 -1 2 0 -6 2 0 1 -2 0 6 -2" />
          <Color color="0.2 0.2 0.2 1 0.058824 0.227451 0.4 1 0.121569" />
        </IndexedQuadSet>
      </Shape>
    </Transform>
  </Scene>
</X3D>
```

index defines sequential quadrilaterals, no -1 sentinel values

**Edit IndexedQuadSet**

containerField: geometry

index: 0 1 2 3 4 5 6 7

ccw  colorPerVertex   
normalPerVertex  solid

OK Cancel Help

**Edit Color**

containerField: color

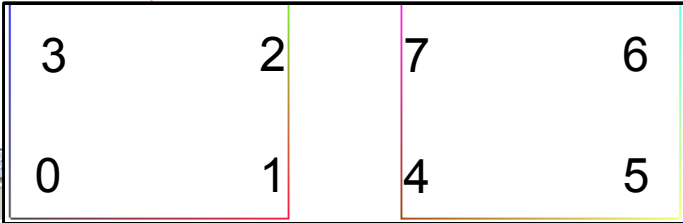
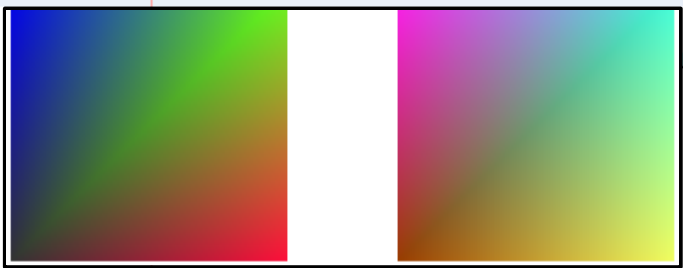
DEF: [ ] USE: [ ]


color array

r	g	b	...
0.2	0.2	0.2	
1.058824	0.227451		
0.4	1.0121569		
0.0007843	0.960784		
0.6	0.235294	0	
0.94902	1.0380392		
0.239216	1.0886275		
1.0121569	0.964706		

+ -

OK Cancel Help



 IndexedQuadSet	<b>[X3D 3.1] IndexedQuadSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.</b> <b>Hint:</b> insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
DEF	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. <b>Hint:</b> descriptive DEF names improve clarity and help document a model.
USE	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children. <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
index	<b>[index: accessType initializeOnly, type MInt32 CDATA #IMPLIED]</b> coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets. Use -1 to separate indices for each polygon.
ccw	<b>[ccw: accessType initializeOnly, type SBool (true false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.
colorPerVertex	<b>[colorPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
normalPerVertex	<b>[normalPerVertex: accessType initializeOnly, type SBool (true false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
solid	<b>[solid: accessType initializeOnly, type SBool (true false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
containerField	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.



# Additional Resources

# Xj3D CAD Filter Converter

**Select Conversion Options**

**Scene results**

3.2 X3D version  Triangle count

ALL Logging level  Embed prototype content

SMALLEST Binary compression method  Set minimum profile

Identity filter (no internal scene-graph changes)

CAD filters of interest

**Filter methods**

1.0 Absolute scale factor  Add bounding boxes  IndexedFaceSet to IndexedTriangleSet

0.001 Floating-point quantization  Center  IndexedFaceSet to TriangleSet

Combine shapes  Index

DEF-USE ImageTexture  Modify viewpoint

Flatten transform branches  Shorten DEF

Generate normal values  Triangulation

**Time-consuming methods**

Re-index  Debug

Reset to defaults

Cancel Continue

# Xj3D converterHelp.txt

Running Xj3D Format Converter converter.bat

CDFFilter - usage: filter [filters] input output [-loglevel type]

[-exportVersion n] [-compressionMethod n ] [-quantization n ] [-upgrade] [filter\_args]

-loglevel type [ALL|WARNINGS|ERRORS|FATAL|NONE]

The minimum level that logs should be written at

-exportVersion n.n

The exported version of the X3D specification to generate

No error checking is performed for invalid version numbers

Assumes 3.1 if not supplied

-compressionMethod [FASTEST|SMALLEST|LOSSY|STRINGS]

Define what sort of compression algorithms should be used

when X3D Binary format is used for the output. Ignored in

all other cases

-quantization n

Positive floating point value that states how much quantization

of values is allowed. Default is 0.001

-upgrade

When declared, any VRML style PROTO content that can be

upgraded to X3D native nodes, will be

Available filters:

MinProfile

CombineShapes

Triangulation

TriangleCountInfo

GenNormals

ShortenDEF

IFSToTS

ReIndex

Debug

DEFUSEImageTexture

AbsScale

Index

Identity

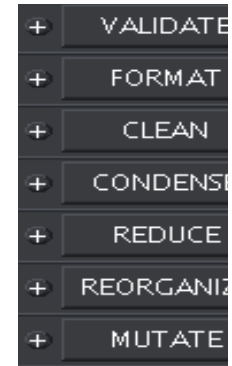
ModifyViewpoint

Center

FlattenTransform

IFSToITS

# Chisel



## Chisel VRML Optimisation Tool

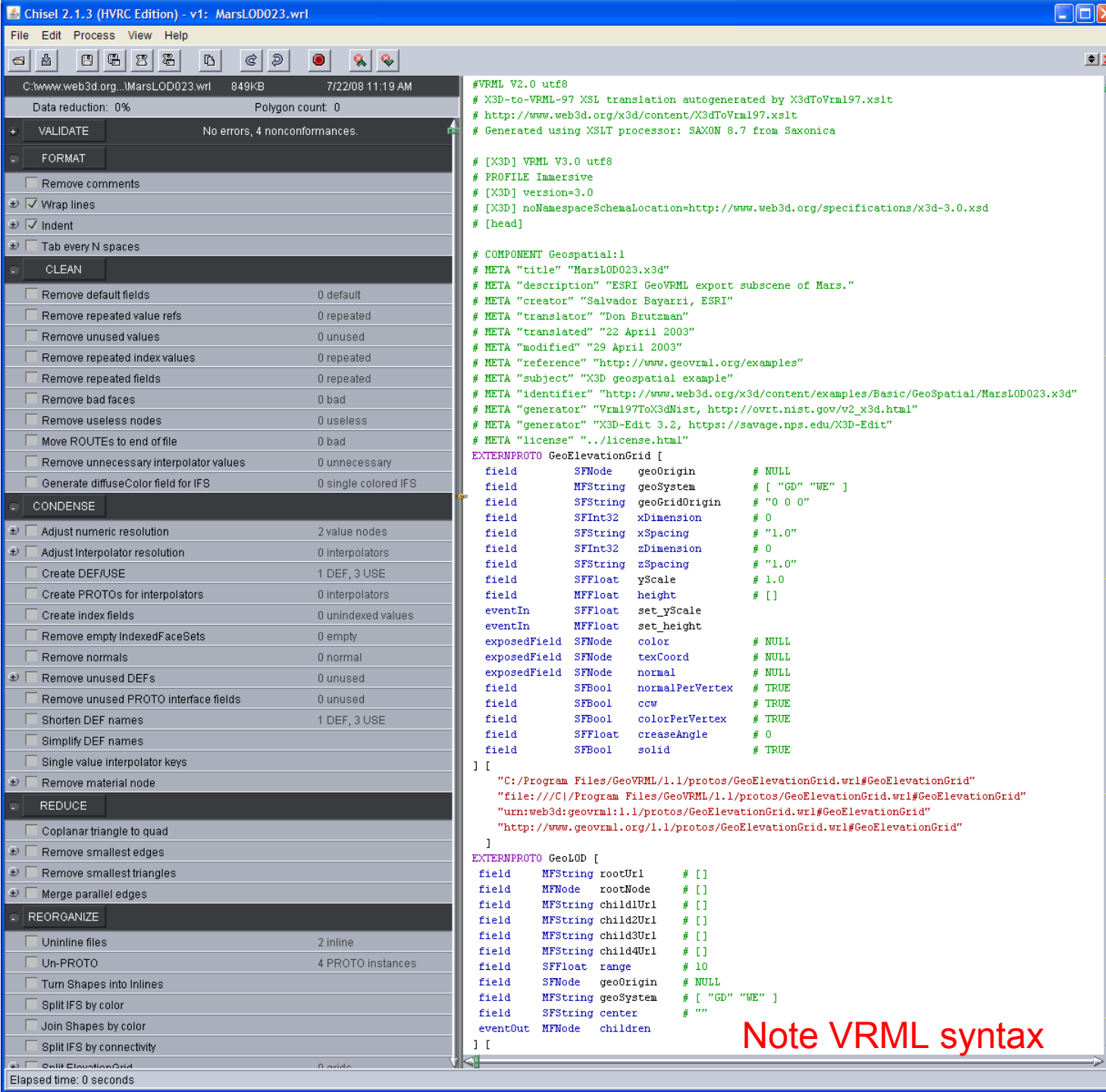
- Features: Validate, Format, Clean, Condense, Reduce, Reorganize, Mutate
- Autoinstaller <http://www2.hrp.no/vr/tools/chisel/install.htm>
- Documentation <http://www2.hrp.no/vr/tools/chisel/doc>

## Open source Java

- Supported by Halden Virtual Reality Centre, Norway
- Originally built by Trapezium, maintained by NIST

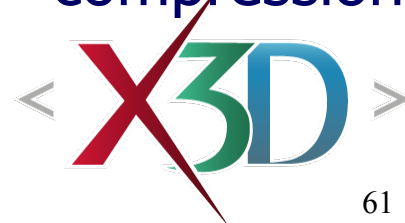
Usage: export to VRML, modify, import back to X3D

Wish list: .x3d handling, X3D-Edit integration



# Chisel Features

- fine-tune
- control of changes
- statistics
- for size,
- polygons
- geometric
- compression
- gzip
- compression



Note VRML syntax

# OpenGL Programming API

OpenGL home has a vast set of specifications and programming resources for the underlying graphics rendering software

- X3D players typically use OpenGL or DirectX to drive the underlying graphics hardware
- With effort, programming tools and algorithms can sometimes be adapted to work with X3D constructs
- <http://www.opengl.org>

OpenGL Programming Guide

- <http://www.glprogramming.com/red>

# Chapter Summary

# Summary: Triangles and Quadrilaterals

## Common fields

- *solid, ccw, colorPerVertex, normalPerVertex*

Review from  
Chapters 2,6

## Normal vectors, Normal node

## Ordered polygonal nodes

- TriangleSet, TriangleFanSet, TriangleStripSet, and QuadSet

## Indexed polygonal nodes

- IndexedTriangleSet, IndexedTriangleFanSet, IndexedTriangleStripSet, and IndexedQuadSet

Pay close attention to detail with these nodes!



# Suggested exercises

Build a Box or a Platonic Solid out of triangles

Write small program to compute polygon values

- Text output of program: viewable .x3d scene file

Apply an image or movie to a billboarded Quad

- Example: Chapter 5, MovieTextureAuthoringOptions.x3d

Convert a simple IndexedFaceSet node into each of TriangleSet nodes to compare differences

- Using pencil + graph paper will help

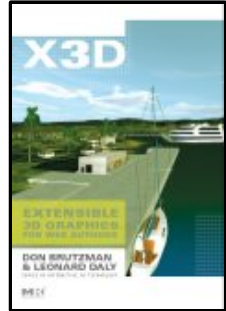
Test Xj3D CAD Filter Converter using X3D-Edit

- IndexedFaceSet to TriangleSet or IndexedTriangleSet

# References

# References 1

*X3D: Extensible 3D Graphics for Web Authors*  
by Don Brutzman and Leonard Daly, Morgan  
Kaufmann Publishers, April 2007, 468 pages.



- Chapter 13, Geometry Nodes part 4:  
Triangles and Quadrilaterals
- <http://x3dGraphics.com>
- <http://x3dgraphics.com/examples/X3dForWebAuthors>

## X3D Resources

- <http://www.web3d.org/x3d/content/examples/X3dResources.html>

# References 2

## X3D-Edit Authoring Tool

- <https://savage.nps.edu/X3D-Edit>

## X3D Scene Authoring Hints

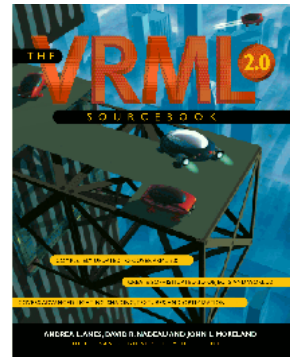
- <http://x3dgraphics.com/examples/X3dSceneAuthoringHints.html>

## X3D Graphics Specification

- <http://www.web3d.org/x3d/specifications>
- Also available as help pages within X3D-Edit

# References 3

*VRML 2.0 Sourcebook* by Andrea L. Ames, David R. Nadeau, and John L. Moreland, John Wiley & Sons, 1996.



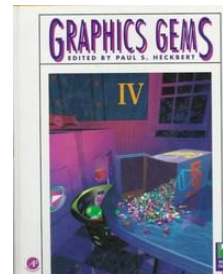
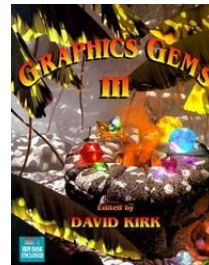
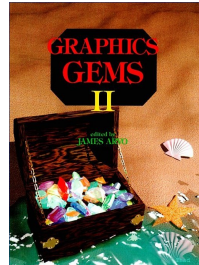
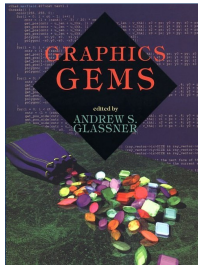
- <http://www.wiley.com/legacy/compbooks/vrml2sbk/cover/cover.htm>
- <http://www.web3d.org/x3d/content/examples/Vrml2.0Sourcebook>
- Chapter 13 - Points Lines Faces covers common fields and related IndexedFaceSet examples

Note: X3D triangle and quadrilateral nodes in this chapter were not yet defined in VRML97

# References 4

*Graphics Gems* book series: many algorithms

- <http://www.graphicsgems.org>



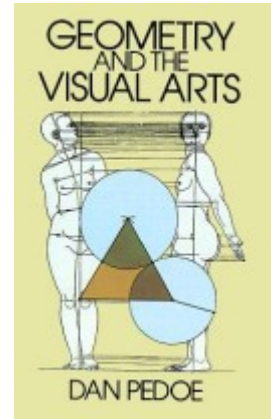
Journal of Graphics Tools (jgt): many algorithms

- <http://jgt.akpeters.com>



# References 5

*Geometry and the Visual Arts*, Daniel Pedoe,  
Dover Publications, 1983.



## Wikipedia

- Triangle strip
- Triangle fan
- Platonic solids
- Archimedean solids

# Contact

**Don Brutzman**

*brutzman@nps.edu*

*http://faculty.nps.edu/brutzman*

Code USW/Br, Naval Postgraduate School

Monterey California 93943-5000 USA

1.831.656.2149 voice



# CGEMS, SIGGRAPH, Eurographics

The Computer Graphics Educational Materials Source(CGEMS) site is designed for educators

- to provide a source of refereed high-quality content
- as a service to the Computer Graphics community
- freely available, directly prepared for classroom use
- <http://cgems.inesc.pt>

*X3D for Web Authors* recognized by CGEMS! 😊

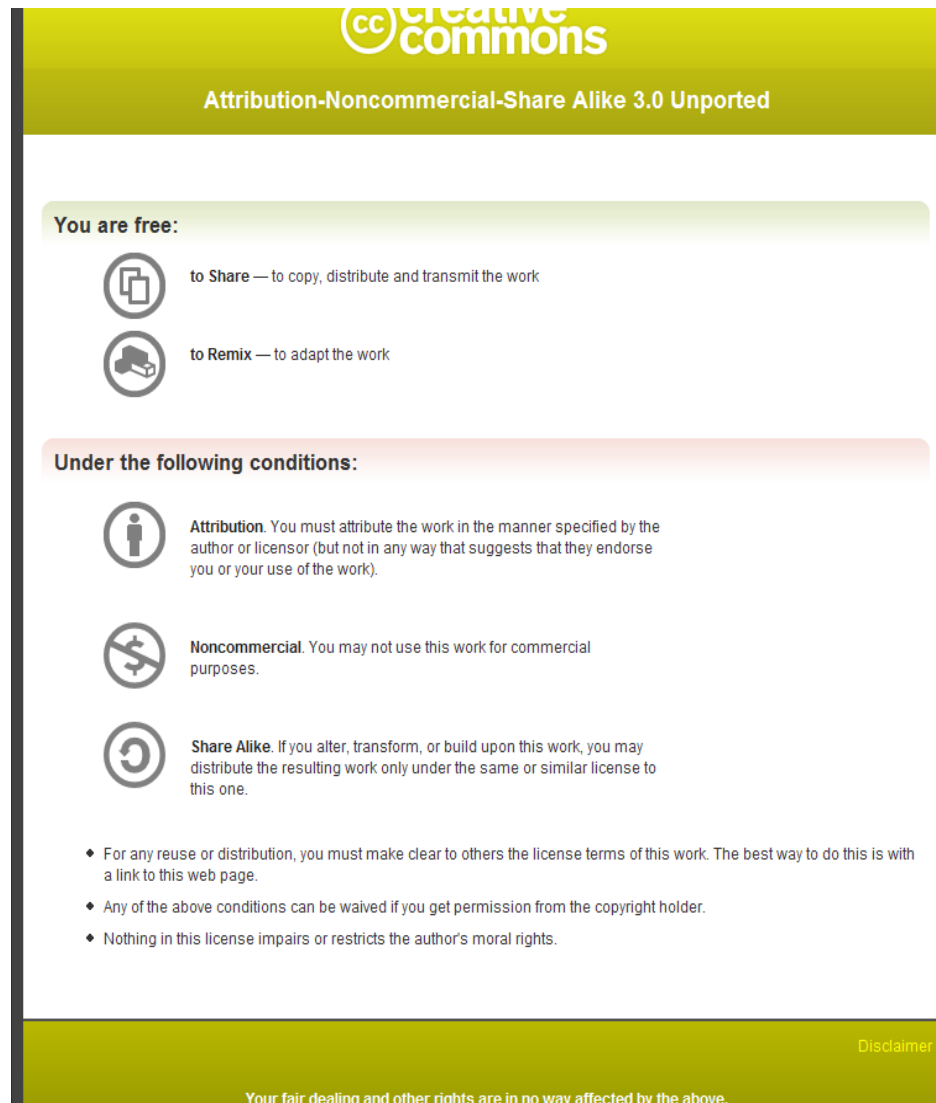
- Book materials: X3D-Edit tool, examples, slidesets
- Received jury award for Best Submission 2008

CGEMS supported by SIGGRAPH, Eurographics



# Creative Commons open-source license

<http://creativecommons.org/licenses/by-nc-sa/3.0>





The image shows a Creative Commons license card for Attribution-Noncommercial-Share Alike 3.0 Unported. The card has a green header with the CC logo and the license name. Below the header, there are two main sections: 'You are free:' and 'Under the following conditions:'. The 'You are free:' section includes icons for 'Share' (a document with an arrow) and 'Remix' (a document with a pencil). The 'Under the following conditions:' section includes icons for 'Attribution' (a person), 'Noncommercial' (a dollar sign with a slash), and 'Share Alike' (a circular arrow). Below these icons are detailed explanations of each condition. At the bottom of the card, there is a disclaimer in a green bar.




**cc** Creative Commons

Attribution-Noncommercial-Share Alike 3.0 Unported

**You are free:**

-  **to Share** — to copy, distribute and transmit the work
-  **to Remix** — to adapt the work

**Under the following conditions:**

-  **Attribution.** You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
-  **Noncommercial.** You may not use this work for commercial purposes.
-  **Share Alike.** If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.

- ◆ For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to this web page.
- ◆ Any of the above conditions can be waived if you get permission from the copyright holder.
- ◆ Nothing in this license impairs or restricts the author's moral rights.

Disclaimer

Your fair dealing and other rights are in no way affected by the above.

# Open-source license for X3D-Edit software and X3D example scenes

<http://www.web3d.org/x3d/content/examples/license.html>

Copyright (c) 1995-2013 held by the author(s). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the names of the Naval Postgraduate School (NPS) Modeling Virtual Environments and Simulation (MOVES) Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# X3D Graphics for Web Authors

## Chapter 13

### Geometry Nodes, Part 4: Triangles and Quadrilaterals

*There is no "royal road" to geometry.*

Euclid, to King Ptolemy I



Reference: Euclid's *Elements*, [http://en.wikipedia.org/wiki/Euclid%27s\\_Elements](http://en.wikipedia.org/wiki/Euclid%27s_Elements)

Attention to detail is essential for understanding these geometry nodes.

# Contents

Chapter Overview and Concepts

X3D Nodes and Examples

Additional Resources

Chapter Summary and Suggested Exercises

References



2

# Contents

Chapter Overview and Concepts

X3D Nodes and Examples

Additional Resources

Chapter Summary and Suggested Exercises

References



# Chapter Overview



## Overview: Triangles and Quadrilaterals

### Common fields

- *solid, ccw, colorPerVertex, normalPerVertex*

Review from  
Chapters 2,6

### Normal vectors, Normal node

### Ordered polygonal nodes

- TriangleSet, TriangleFanSet, TriangleStripSet, and QuadSet

### Indexed polygonal nodes

- IndexedTriangleSet, IndexedTriangleFanSet, IndexedTriangleStripSet, and IndexedQuadSet

Pay close attention to detail with these nodes!



[back to Table of Contents](#)

# Concepts



Many concepts are common to Chapter 6, Geometry part 2: Points Lines & Polygons. Some points are also covered in Chapter 2 Geometry Primitives.

## Many different geometry nodes

An excellent aspect of X3D is that there are many different ways to create geometry

- Chapter 2, Geometry Primitives
- Chapter 6, Points, Lines and Polygons
- Chapter 10, Geometry2D Nodes
- Chapter 13, Triangles and Quadrilaterals

These are all handled consistently inside a Shape node with corresponding Appearance



X3D consistency is pretty powerful. All the other ideas and implementation details for geometry nodes shown in prior chapters remain consistent for the triangle and quadrilateral nodes presented in this chapter.

Also of interest is the X3D Non-Uniform Rational B-Spline (NURBS) component.

## Motivation

Triangle and Quadrilateral nodes do not need to be tessellated (turned into triangles) by X3D viewers because they are already in that form

Polygons are defined in ways that can be directly passed to the underlying graphics hardware

Triangle nodes are perhaps the most low-level or fundamental nodes since they can identically represent any other polygonal node in X3D, if vertex definitions and connectivity ordering are properly expressed



Indeed the typical necessary task performed by X3D viewers for the other “simpler, higher-level” geometry nodes is to convert them into one of the triangular representations. Once that tessellation is performed, the underlying graphics rendering API and hardware can efficiently render them.

Although the triangle and quadrilateral nodes in this chapter are often called “low level,” the term indicates that they define data in a form that matches the graphics hardware underlying the graphics software. However, for authors, properly and precisely defining correct polygon data for these nodes can be quite tedious and error prone. Thus the “higher level” nodes defined in the other geometry chapters are actually easier to work for most modeling tasks.

An interesting feature for some X3D tools might be to reduce all higher level geometry nodes into triangle nodes. Likely this will increase file size, but it ensures that the tessellation is performed exactly as the author expects. Such a step is also a good way to obfuscate (hide) the parameters of a carefully constructed shape, protecting an author's intellectual property and thus making downstream modification more difficult. If geometry compression is also applied, make sure that quality has not degraded.

Of some historical interest is that the original design of these X3D nodes was checked closely by X3D browser builders, ensuring that they are a good match for both the OpenGL and DirectX rendering application programming interfaces (APIs). In this way, if an author wants, highly efficient X3D geometry can be defined that directly maps to the underlying hardware without requiring browser tessellation.

# Triangles

Triangles are the primary low-level geometry construct used by graphics software, hardware

- More complex shapes are reduced to triangles by the rendering software
- A triangle is always planar, allowing the material appearance to fill it

Sometimes quadrilaterals are used, but problem is that values might be non-coplanar due to roundoff (or authoring) error

- Which means that filling in material is ill defined, and not properly or repeatably renderable

Graphics hardware is highly optimized to favor triangles, performing something very simple, many times, extremely quickly.

Authors can guarantee that quadrilaterals remain planar by defining them within horizontal or vertical planes. In other words, one of the x, y or z coordinates are equal for all four vertices (for example, all four coordinate values have  $z=0$ , only varying with respect to their x and y coordinates). In that way, if there is any roundoff due finite floating-point precision, the truncated values for each point in the quadrilateral will be equal.

## Single-sided polygons

Graphics engines always prefer simplicity in order to achieve maximum run-time performance

- Top 3 considerations for graphics hardware: performance, performance, performance!

Single-sided polygons take about half the time to draw than double-sided polygons

- So if authors can arrange geometry so that only one side is ever visible to user, can go single-sided
- Technical term: *backface culling*
- Efficiency is rationale for many X3D default values
- Example: default setting is *solid='true'*
- Debugging hint: set *solid='false'* to show both sides

Important technique: set *solid='false'* to show both sides of all polygons, helping to expose inside-out or missing triangles. The performance handicap is typically slight, especially compared to the alternative: the user is missing unseen model parts!

## Common field: *solid*

In 3D graphics, all triangles have 2 sides

- Graphics term: backface culling only draws front sides

The *solid* field defines whether a geometry node has an inside or not, with a default value of true

- *solid*='true' means do not render (draw) the inside
- *solid*='false' means render both inside and outside

This approach reduces the number of polygons needing to be drawn, thus improving performance

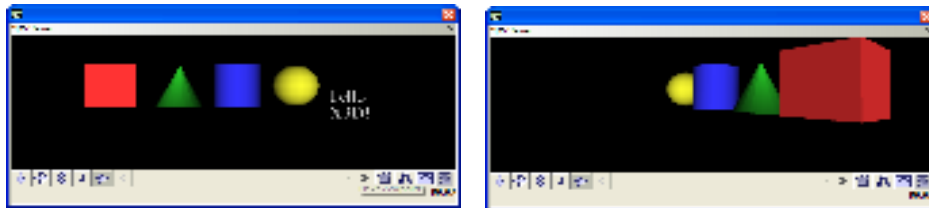
Confusing if user gets lost inside invisible geometry

- **Hint:** set *solid*='false' to draw both sides

## Common field: *solid*

To see an example of 'solid' geometry, rotate the GeometryPrimitives.x3d scene by 180 degrees

- Once rotated, the first four shapes remain visible, but the Text node disappears
- This is because *solid='true'* by default, so the reverse side of text is not drawn by default



Here we are still using the example

<http://www.x3dbook.com/examples/X3dForWebAuthors/Chapter02-GeometryPrimitives/GeometryPrimitiveNodes.x3d>

The original scene (on the left) is rotated about 150 degrees to the right. To do this, click on the left center of the screen and drag the mouse (pointer) to the right.

You need to have browser navigation in EXAMINE mode for this view rotation to work. In Xj3D, which is used in X3D-Edit, the EXAMINE mode icon is the stylized eye (fifth button on the lower left as shown here).

## Normal vectors

The *normal* vector is perpendicular to the face,  
pointing away from the centroid of polygon

Direction of normal vector defined by order of  
points defining the polygon and right-hand rule

- Align curvature of fingers to match polygon vertex points in order: indices 0, 1, 2 ...
- Thumb points in direction of (positive) normal, which is the front-facing side
- Negative normal thus points in direction of backface

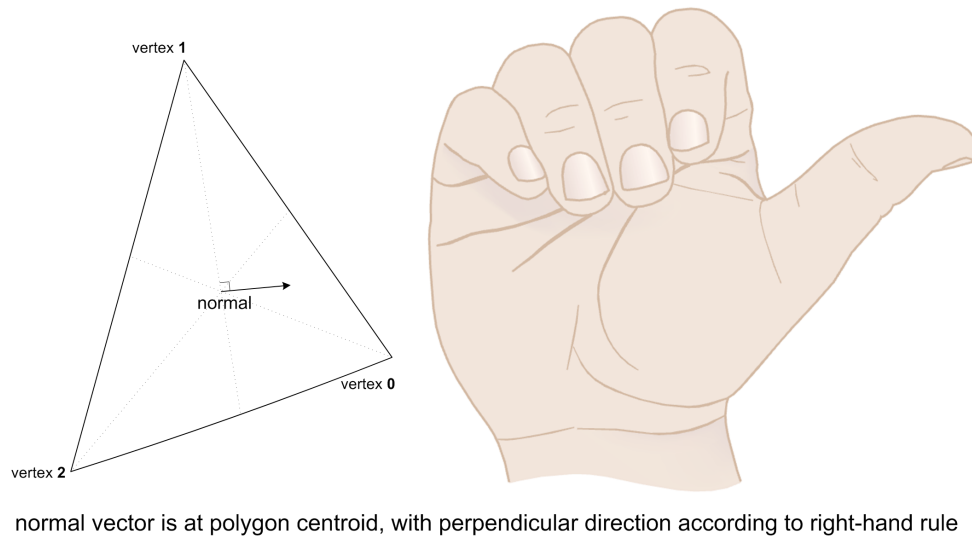
Normal vectors are only pertinent to polygons, not polyline segments or point sets. The Normal node is covered in Chapter 13, Triangles and Quadrilaterals (this chapter).

The NormalInterpolator node is covered in Chapter 7, Event Animation.



Review from  
Chapter 6

## Right-hand rule for polygon normals



This rule also applies to quadrilaterals and polygons that have more vertices.

Vertices that are not coplanar are degenerate, and can lead to erroneous normal computation and unpredictable rendering results.

Normal vectors are only pertinent to polygons, not polyline segments or point sets. The Normal node is covered in Chapter 13, Triangles and Quadrilaterals.

Note that order of vertices must match the curl of the right hand. If not, you are looking at the triangle backface and pointing in the opposite direction, rather than the (positive) normal direction.

## Common field: *ccw*

*ccw* (counter clockwise) indicates whether default normal direction of polygons is counterclockwise (default) or clockwise

- *ccw*='true' is right-hand rule
- *ccw*='false' is opposite

Hint: can correct some opposite-rendering geometry by reversing *ccw* value, rather than reordering all coordinates or indices

- Big time saver for some imports

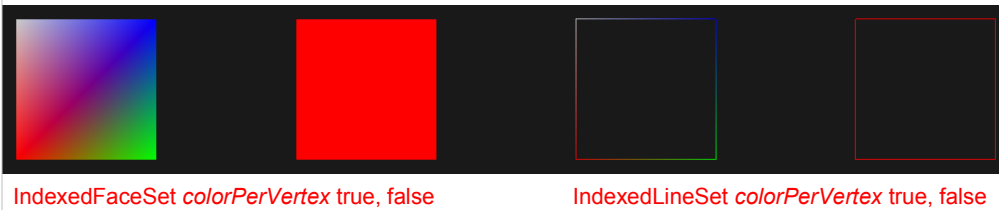
Look at a circular clock face, and apply the right-hand rule at the center with the thumb pointing out from the wall. The curvature of the fingers is counterclockwise.

Note that the counterclockwise field *ccw* has type SFBool (boolean).

## Common field: *colorPerVertex*

*colorPerVertex* indicates whether contained color values are applied to each vertex point (default), or to each polygonal face

- *colorPerVertex*='true' requires that # colors must equal # points
- *colorPerVertex*='false' requires that # colors must equal # polygons



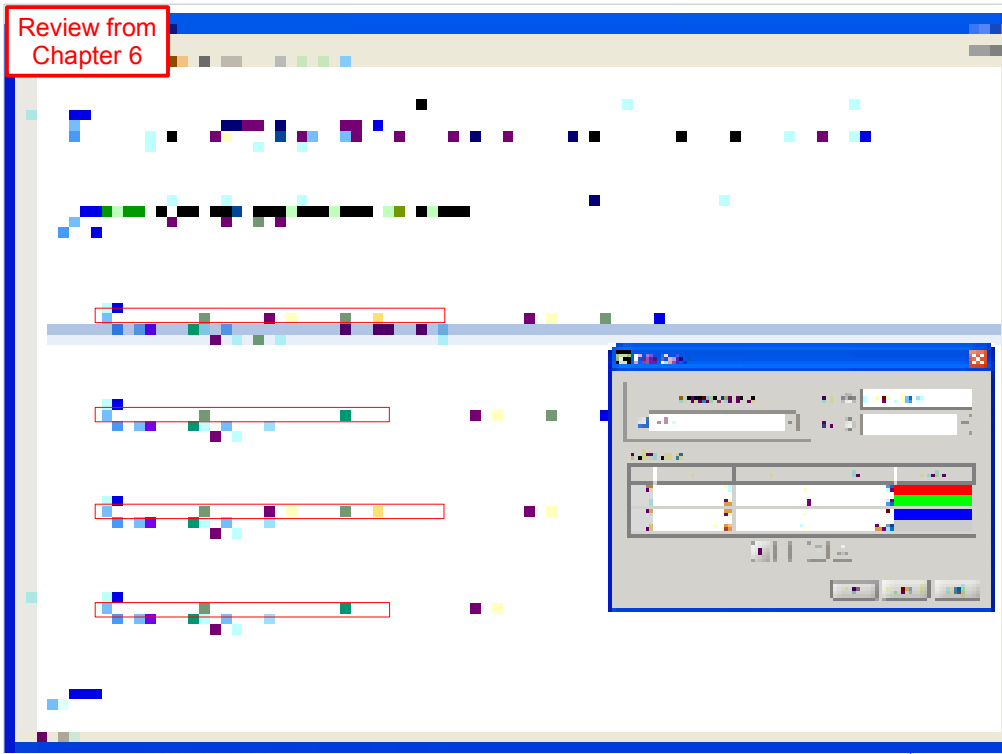
In this context, read # as 'number of'

*colorPerVertex* type is SFBool (boolean).

Given either value for *colorPerVertex*: counting the number of colors, along with the number of points or polygons, then making sure that those numbers are consistent, is an important correctness check.

*X3D for Web Authors*, Figure 6.1, p. 160

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ColorPerVertexExamples.x3d>



<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ColorPerVertexExamples.x3d>

In addition to illustrating *colorPerVertex* effects, this example shows how Coordinate and Color nodes can be shared by both IndexedFaceSet and IndexedLineSet.

Also note how each set of *coordIndex* and *colorIndex* fields form a closed loop, both starting and ending with index 0 (prior to the end-of-polygon/end-of-polyline sentinel value -1).

ColorRGBA can be used instead of Color if alpha channel (alpha = 1 - transparency) is also needed for polygons or vertices.

## Common field: *normalPerVertex*

*normalPerVertex* indicates whether contained normal values are applied to each vertex point (default), or to each polygonal face

- *normalPerVertex*='true' requires that # normals must equal # points
- *normalPerVertex*='false' requires that # normals must equal # polygons

In this context, read # as 'number of'

Note that the *normalPerVertex* type is SFBool (boolean).

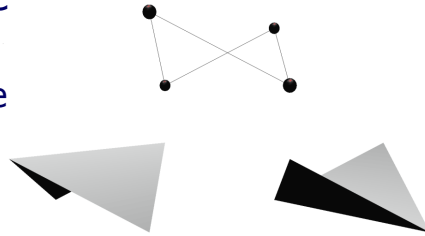
Given either value for *normalPerVertex*: counting the number of normals, along with the number of points or polygons, then making sure that those numbers are consistent, is an important correctness check.

Normal vectors are only pertinent to polygons, not polyline segments or point sets.

# Nonplanar quadrilaterals

Nonplanar quadrilaterals are ambiguously defined

- Can be triangulated in more than one way
- Can produce unpredictable results, rather than a consistent model for users



Nonplanar vertices in a single quadrilateral are thus undesirable and need to be avoided

*X3D for Web Authors*, Figure 13.1, p. 360.

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/NonplanarPolygons.x3d>

Defining nonplanar vertices together in a single polygon is undesirable because the triangulation possibilities are inherently ambiguous and unresolvable. Depending on which tessellation split is performed across the diagonal of the quadrilateral polygon, two quite-different pairs of triangles can be produced from a single set of four nonplanar points.

## Common characteristics for these nodes

### Triangle and quad (quadrilateral) nodes

- Vertex values are defined in child `Coordinate` or `CoordinateDouble` node
- Colors are defined in child `Color` or `ColorRGBA` node
- May also have child `Normal` and `TextureCoordinate` (or `TextureCoordinateGenerator`) nodes

### Common attributes

- *ccw*, *solid*, *colorPerVertex*, *normalPerVertex*
- Indexed nodes include *index* integer array
- Animation possible using *set\_index* field
- **Caution:** some nodes use -1 sentinel *index* value but many don't, so always check indexing rules closely

## Distinct characteristics for each node

Color values follow the same indexing scheme as corresponding Coordinate point values

- TriangleSet, QuadSet: ordered sequence of values
- TriangleFanSet: *fanCount*
- TriangleStripSet: *stripCount*
- IndexedTriangleSet, IndexedQuadSet, IndexedTriangleFanSet, IndexedTriangleStripSet: all use common *index* array

Identical indexing correspondences are used for applying contained Normal, TextureCoordinate and TextureCoordinateGenerator node values

- No *colorIndex*, *normalIndex*, or *texCoordIndex* fields

Using values from the Color, Normal and TextureCoordinate nodes in the same order as the values from the Coordinate node helps hardware implementations operate quickly. Ordering values in a scene can thus be considered a form on preprocessing. Such preparation is usually handled by the X3D viewer anyway.



[back to Table of Contents](#)

## X3D Nodes and Examples



22

## Normal node

Normals are perpendicular vectors that are used in X3D lighting equations to shade geometry

Normals are typically computed automatically

- Since they can be unambiguously calculated from polygon coordinates
- Reduces file size and bandwidth requirements

Authors (or at least authoring tools) can precompute normal values and include them

- Possible reduction in load time
- Can enable application of special effects

Such precomputation of normal values is unusual and typically reserved for special effects. It is much easier to let the X3D player compute normal values.

## Effects of normal vectors

Recall that polygon lighting is computed by:

- projecting light rays from each light source, then
- reflecting them off intermediate polygons, until
- colored, modified light reaches user view

Incident and reflected light angles over the surface plane are also equal about the perpendicular

- Hence normal vector determines angle of reflection
- Direct straight-line reflections have greatest intensity
- Varying the direction of normal vectors can change the perceived brightness of each reflection, thus providing a special effect

## Normal vectors and lighting computations

Varying normal vector varies reflection intensity  
since brightest reflection is from smallest angle



Handwritten text below the diagram, likely describing the relationship between the normal vector and the viewer's direction, and how it affects the reflection intensity.

*X3D for Web Authors*, Figure 13.2, p. 362. Varying a normal vector toward (or away from) a light source increases (or reduces) the computed intensity of a viewed pixel.

## Unit normalization of normal vectors

Normalization is the process of changing the magnitude of a vector to have unit length

- Not same “normal” as perpendicular normal vector

Normal vector values should be unit normalized

- Allowing faster processing by graphics hardware
- Zero-magnitude vectors are degenerate, erroneous

Procedure:

- divide each vector component by magnitude

$$\text{Magnitude } r = \sqrt{(x^2 + y^2 + z^2)} \quad x' = x/r; y' = y/r; z' = z/r;$$

web|3D  
CONSORTIUM

Normalized vector  $N' = (x', y', z')$



26

Two different terms are used together here:

- *Normal vectors* are perpendicular to a planar surface
- *Unit normalization* ensures that a vector has unit magnitude (length = 1)

## Normal-node hints and warnings

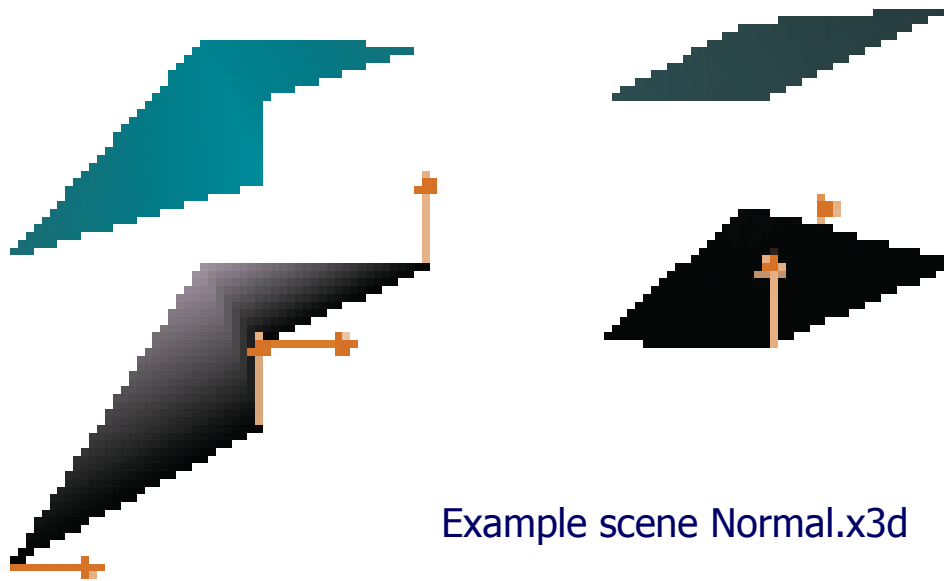
Adding normal values can reduce load time through reduced computation, but

- Computation is already quite fast
- File loading may take longer due to larger file size
- Network delay may be longer due to larger file size

Precomputing and animating normals may aid animations if

- Special rendering effects are desired
- Recomputation of normals becomes necessary due to changes in complex mesh geometry

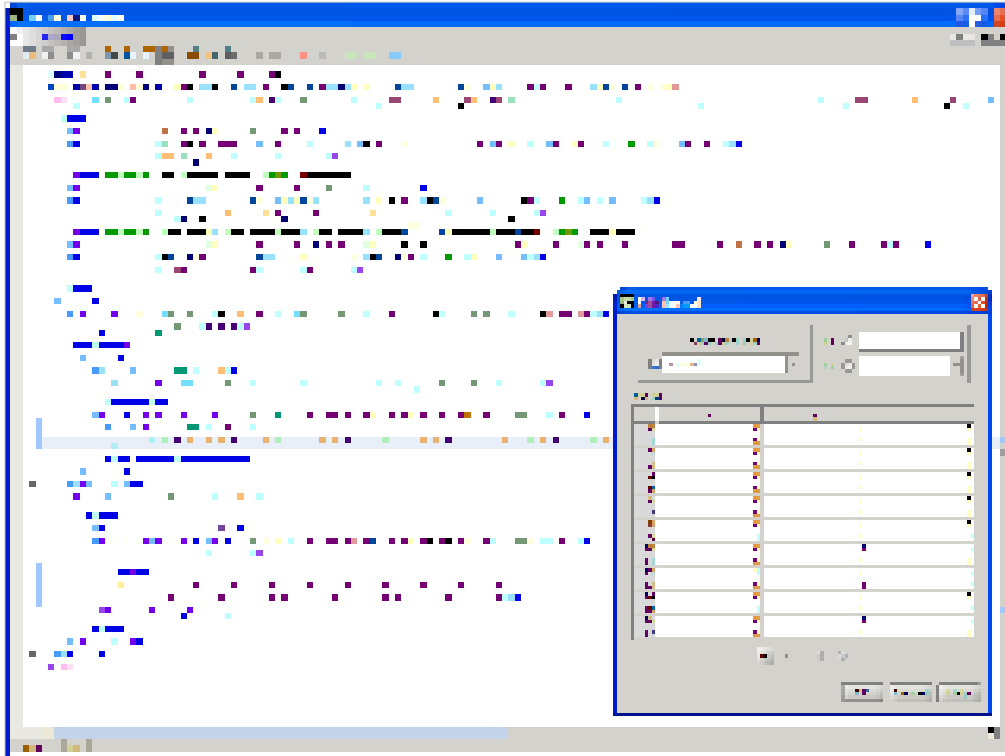
## Example modification of normals



<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/Normal.x3d>


*X3D for Web Authors*, Figure 13.3, p. 363. Modification of normals in the example scene demonstrates special effects that vary shading.

Note that there is no animation in this example, but it is a special effect nevertheless.



<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/Normal.x3d>



 <b>Normal</b>	Normal defines a set of 3D surface-normal vectors. Normal values are optional perpendicular directions, used per-polygon or per-vertex for lighting and shading. Hint: used by IndexedFaceSet and ElevationGrid.
DEF	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
vector	<b>[vector: accessType inputOutput, type MFVec3f CDATA #IMPLIED]</b> set of unit-length normal vectors, corresponding to indexed polygons or vertices.
containerField	<b>[containerField: NMTOKEN "normal"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#Normal>

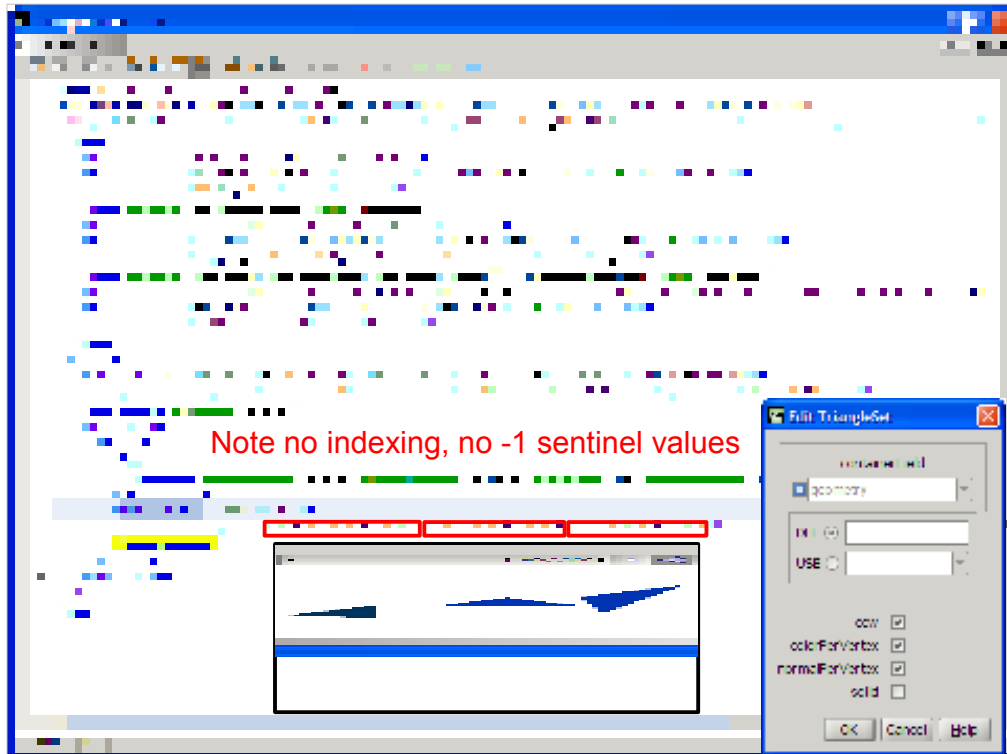
# TriangleSet node

Most basic node, representing a set of triangles

- Vertex values are defined in child Coordinate or CoordinateDouble node
- Colors are defined in child Color or ColorRGBA node
- See Common Characteristics

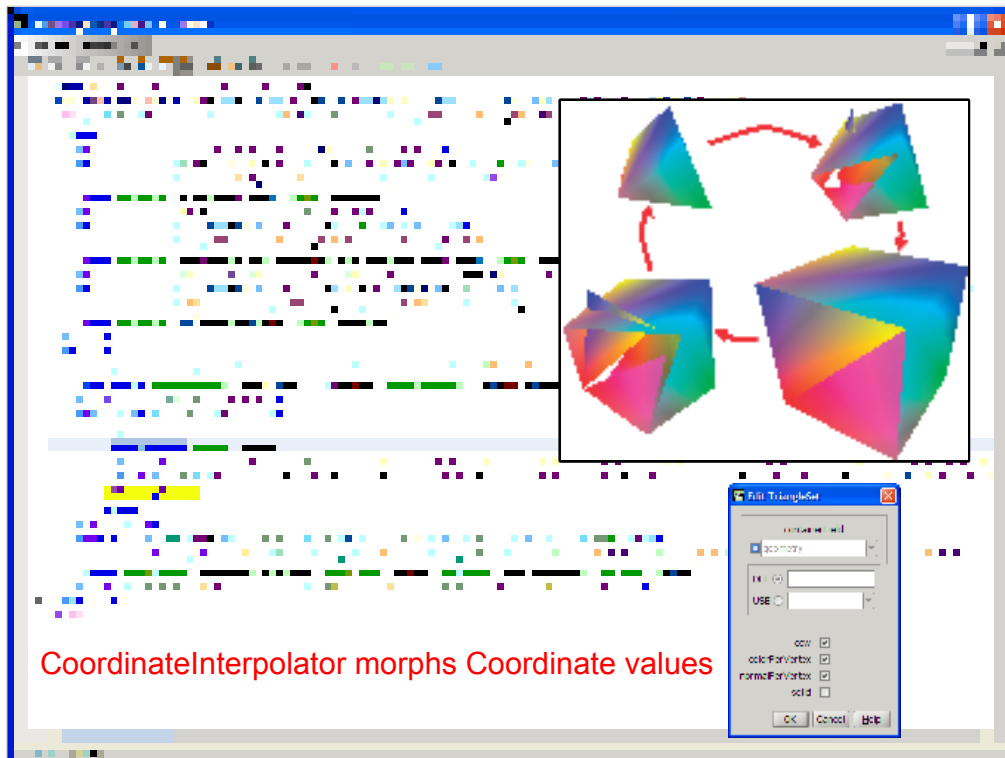
No counting or indexing, no -1 sentinel values

- Each a-b-c sequence of *point* triplets in Coordinate node defines vertices of an independent triangle
- Coincident vertex values must be repeated, and so efficiency is reduced for large Coordinate nodes



<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/TriangleSet.x3d>

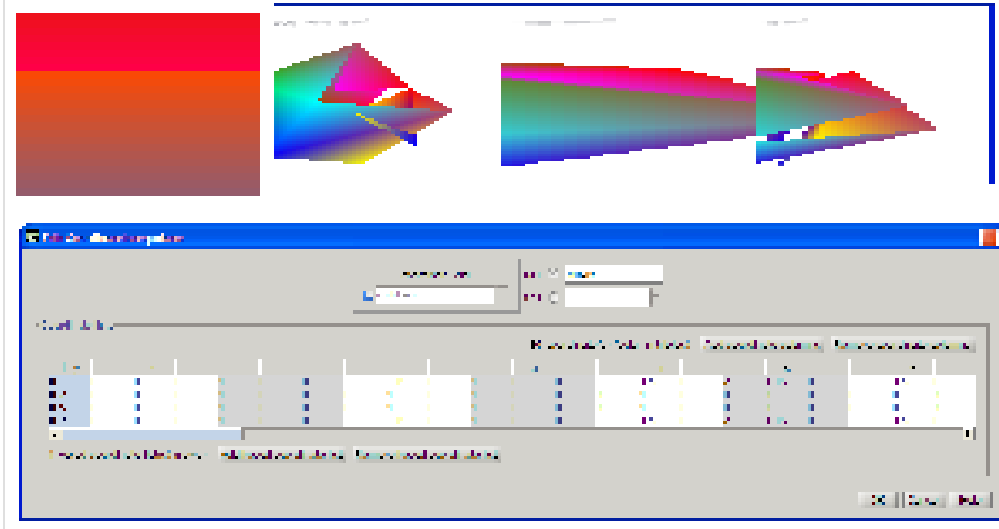
*X3D for Web Authors*, Figure 13.4, p. 365. TriangleSet consisting of three triangles and nine separate vertices.



<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/MorphingTriangleSet.x3d>

*X3D for Web Authors*, Figure 13.5, p. 365. Morphing a pyramid into a cube by animating a TriangleSet. This sequence of images shows (clockwise from top left) an initial pyramid, intermediate pyramid to cube, cube, and intermediate cube to pyramid.

# MorphingTriangleSet example




<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/MorphingTriangleSet.x3d>

Alternate viewpoint screen snapshots (two box, two pyramid) for same example.

Also shown: CoordinateInterpolator editor holding arrays of replacement coordinates.

*X3D for Web Authors*, Figure 13.5, p. 365. Morphing a pyramid into a cube by animating a TriangleSet. This sequence of images shows (clockwise from top left) an initial pyramid, intermediate pyramid to cube, cube, and intermediate cube to pyramid.

 <b>TriangleSet</b>	<b>TriangleSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.</b> Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
<b>DEF</b>	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
<b>USE</b>	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
<b>ccw</b>	<b>[ccw: accessType initializeOnly, type SBool (true/false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.
<b>colorPerVertex</b>	<b>[colorPerVertex: accessType initializeOnly, type SBool (true/false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
<b>normalPerVertex</b>	<b>[normalPerVertex: accessType initializeOnly, type SBool (true/false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
<b>solid</b>	<b>[solid: accessType initializeOnly, type SBool (true/false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
<b>containerField</b>	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
<b>class</b>	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#TriangleSet>

# TriangleFanSet node

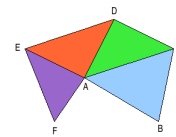
Efficient way to generate a group of triangles that share a single common vertex

- Fan as in "hand-held fan"
- Each triangle created from latest point, preceding point, and initial center point



Indexing via *fanCount* array, no -1 sentinel values

- Each individual fan has  $(fanCount[i]-2)$  triangles
- Each  $a,b,c,\dots,n$  set of Coordinate *point* values define independent triangular fan
- Efficient for small pieces of geometry
- Coincident vertex values must be repeated, and so efficiency is reduced for large Coordinate nodes



This structure is reasonable efficient because, after the first three points are defined, only one point is needed for each additional triangle. However, for large geometry, only so many triangles can be added to a single fan before another fan is needed.

Any convex polygon may be triangulated (tessellated) into a single fan by selecting any vertex as the fan center. Challenge: can you prove (or disprove) this statement?

The length of the *fanCount* array defines how many fans are generated.

The sum of the *fanCount* array elements must not exceed the number of Coordinate points contained.

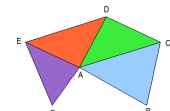
Perhaps curiously, the value of *colorPerVertex* is always ignored and treated as false.

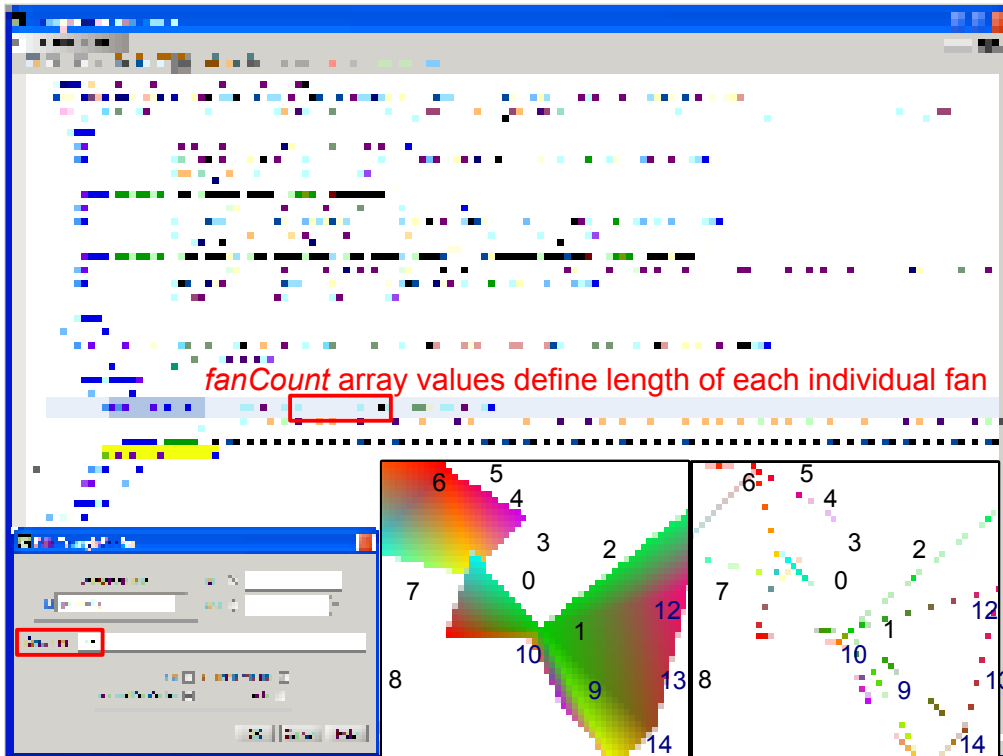
It is possible to create a fan that is self-intersecting. (Not very pretty though.)

Hand-held fan image published with permission from [http://en.wikipedia.org/wiki/Image:Non\\_electric\\_fan\\_aka\\_soljader.jpg](http://en.wikipedia.org/wiki/Image:Non_electric_fan_aka_soljader.jpg)



Triangle fan image published with permission from [http://en.wikipedia.org/wiki/Triangle\\_fan](http://en.wikipedia.org/wiki/Triangle_fan)






<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/TriangleFanSet.x3d>

*X3D for Web Authors*, Figure 13.6, p. 367. TriangleFanSet example showing two fan sets. Each set is colored by vertex.

Note that vertex 2 of the second fan set is not labeled, since it is obscured by triangle 0-7-8 in the first fan set.



 <b>TriangleFanSet</b>	<b>TriangleFanSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.</b> <b>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</b>
<b>DEF</b>	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. <b>Hint: descriptive DEF names improve clarity and help document a model.</b>
<b>USE</b>	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. <b>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</b> <b>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</b>
<b>fanCount</b>	<b>[fanCount: accessType initializeOnly, type MFloat32 CDATA #IMPLIED]</b> (3..+infinity) fanCount array provides number of vertices in each fan.
<b>ccw</b>	<b>[ccw: accessType initializeOnly, type SBoolean (true/false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. <b>Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.</b>
<b>colorPerVertex</b>	<b>[colorPerVertex: accessType initializeOnly, type SBoolean (true/false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
<b>normalPerVertex</b>	<b>[normalPerVertex: accessType initializeOnly, type SBoolean (true/false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
<b>solid</b>	<b>[solid: accessType initializeOnly, type SBoolean (true/false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning: default value true can completely hide geometry if viewed from wrong side!</b>
<b>containerField</b>	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
<b>class</b>	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

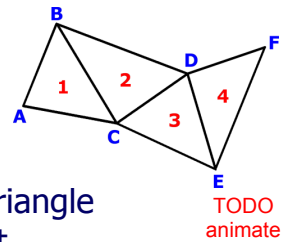
<http://www.web3d.org/x3d/content/X3dTooltips.html#TriangleFanSet>

TODO: add *fanCount* warnings to tooltip

# TriangleStripSet node

Efficient way to generate geometry as a long strip of triangles

- Diagram of four triangles 1, 2, 3, 4, with vertices A, B, C, D, E, and F
- After first triangle, each subsequent triangle is defined by addition of a single point



Indexing via *stripCount* array, no -1 sentinel values

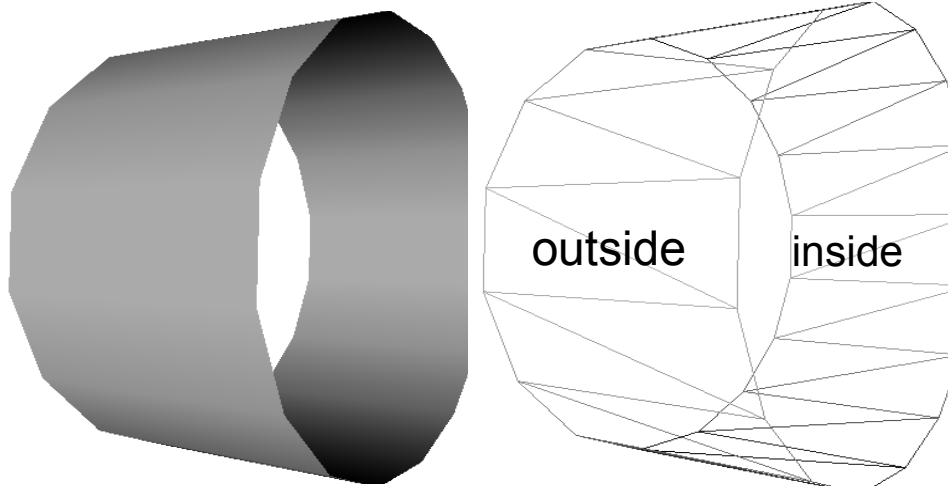
- Each individual strip has  $(stripCount[i]-2)$  triangles
- Each a,b,c,...,n sequence of Coordinate *point* values defines an independent triangular strip
- Coincident vertex values must be repeated, and so efficiency may be reduced for large Coordinate nodes

Large complex geometry may be split into several strip sets, which nevertheless repeat common vertices and thus increase file size.

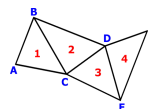
The value of *colorPerVertex* is always ignored and treated as false.

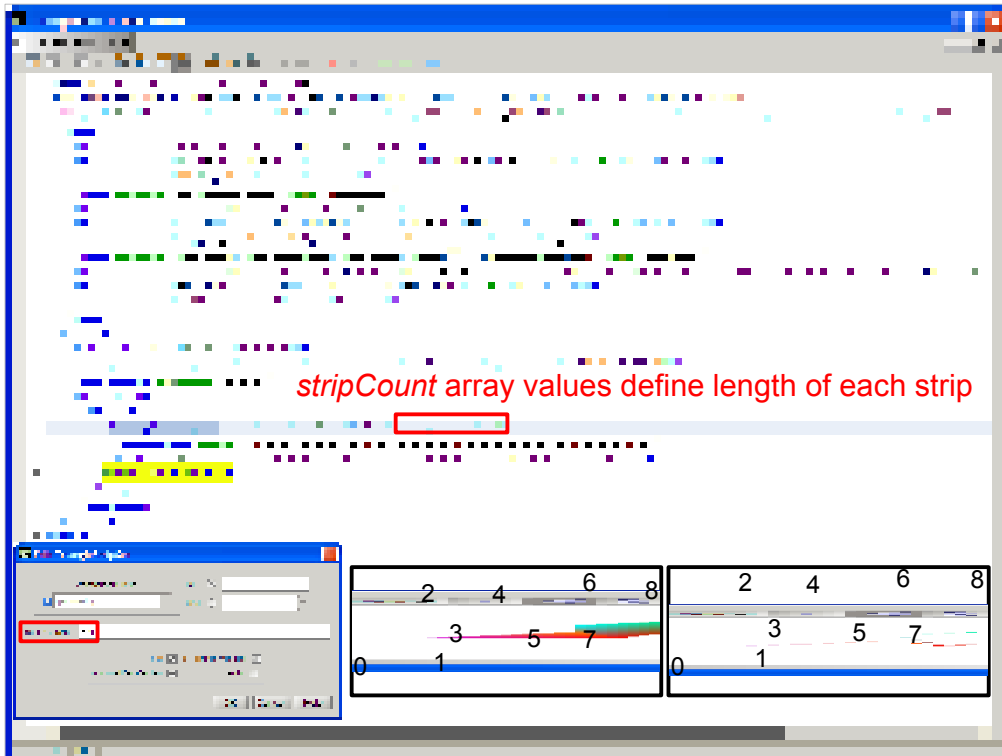
It is possible to create a strip that is self-intersecting.

Example strip set: look at Cylinder sides in wireframe mode.




Triangle strip image published with permission from [http://en.wikipedia.org/wiki/Triangle\\_strip](http://en.wikipedia.org/wiki/Triangle_strip)





<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/TriangleStripSet.x3d>

*X3D for Web Authors*, Figure 13.7, p. 369. TriangleStripSet example using the same nine vertices used for TriangleSet example.

 <b>TriangleStripSet</b>	TriangleStripSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node. Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
DEF	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring all other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
stripCount	<b>[stripCount: accessType initializeOnly, type MInt32 CDATA #IMPLIED]</b> (3..+infinity) stripCount array provides number of vertices in each strip.
ccw	<b>[ccw: accessType initializeOnly, type SBool (true/false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.
colorPerVertex	<b>[colorPerVertex: accessType initializeOnly, type SBool (true/false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
normalPerVertex	<b>[normalPerVertex: accessType initializeOnly, type SBool (true/false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
solid	<b>[solid: accessType initializeOnly, type SBool (true/false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
containerField	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#TriangleStripSet>

# QuadSet node

Defines a set of quadrilaterals (rectangles)

- Coordinate *point* value sequences need to be planar

No counting or indexing, no -1 sentinel values

- Each a-b-c-d set of *point* triplets in Coordinate node defines corners of an independent quad
- Coincident vertex values must be repeated, and so efficiency is reduced for large Coordinate nodes

Hint: must set CADGeometry component level 1

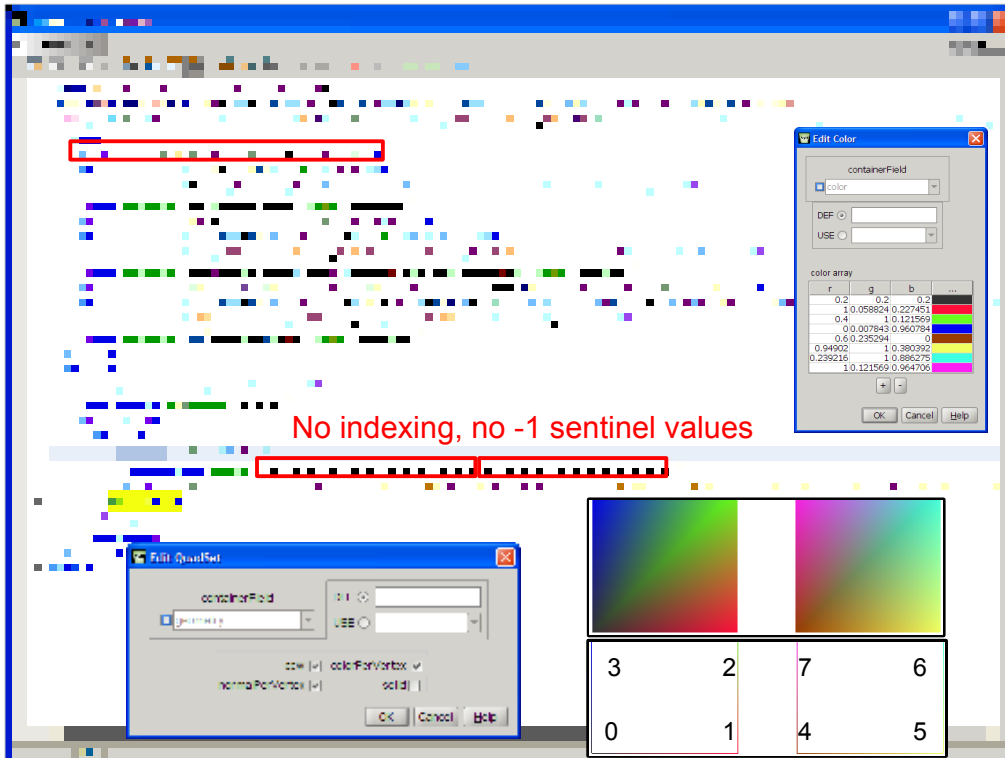


The value of *colorPerVertex* is always ignored and treated as false.


QuadSet does not include the *convex* field. Software renderers (or perhaps more accurately, the underlying graphics hardware cards) are expected to be able to handle concave quadrilaterals satisfactorily, since concavity of a 4-sided polygon is easier to compute than a concave n-sided polygon.

Any quadrilateral can be split into a pair of triangles. Two different splits are possible, matching the two internal diagonals. This is illustrated on the [Nonplanar quadrilaterals](#) slide.

The QuadSet node is somewhat similar to the Rectangle2D node, defined in Chapter 10 Geometry2D. The primary difference is that QuadSet can be out of the X-Y plane and is defined using 3D coordinates.



<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/QuadSet.x3d>

 <b>QuadSet</b>	[X3D 3.1] QuadSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node. Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring all other attributes and children. Hint: USING other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
ccw	[ccw: accessType initializeOnly, type SBool (true false) "true"] ccw = counterclockwise: ordering of vertex coordinates orientation. Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.
colorPerVertex	[colorPerVertex: accessType initializeOnly, type SBool (true false) "true"] Whether Color node is applied per vertex (true) or per polygon (false).
normalPerVertex	[normalPerVertex: accessType initializeOnly, type SBool (true false) "true"] Whether Normal node is applied per vertex (true) or per polygon (false).
solid	[solid: accessType initializeOnly, type SBool (true false) "true"] Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
containerField	[containerField: NMTOKEN "geometry"] containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	[class CDATA #IMPLIED] class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#QuadSet>

# IndexedTriangleSet node

Indexed version of TriangleSet node

- Triangles, triangles and nothing but more triangles!

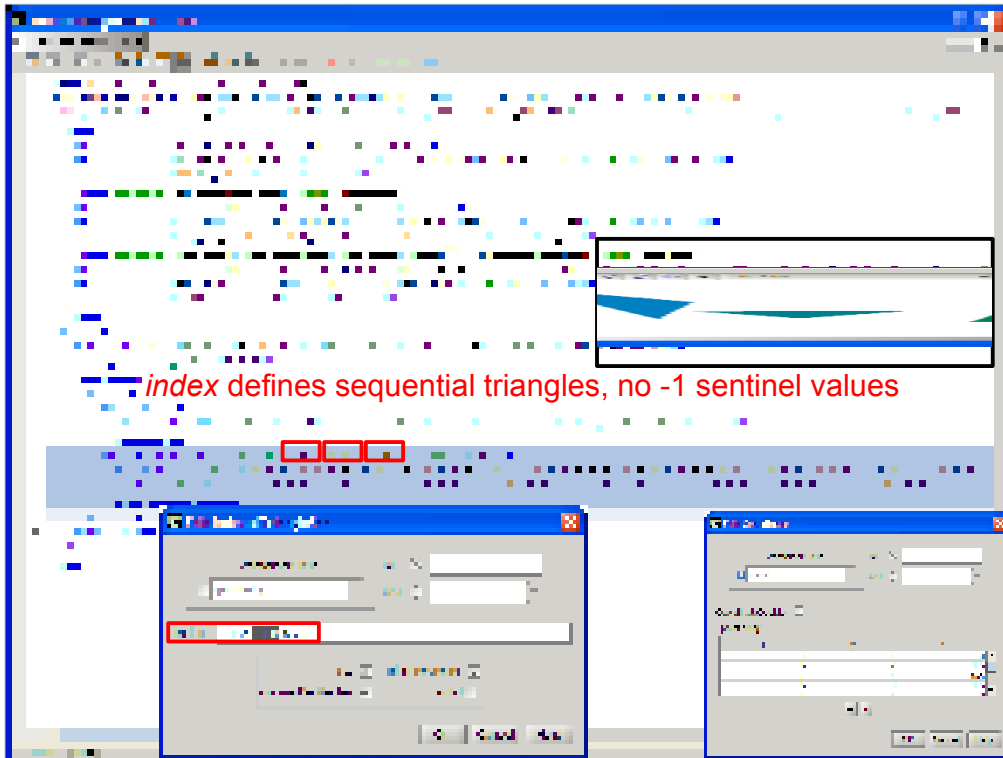
Sequential triplets of *index* array values define triangles, so no -1 sentinel values are inserted

- Each sequential triplet from index array selects 3 Coordinate point values as independent triangle
- Thus no sentinel needed to distinguish triangles
- Efficient: each x-y-z point only needs to be defined once, since indexing can reuse it at any time



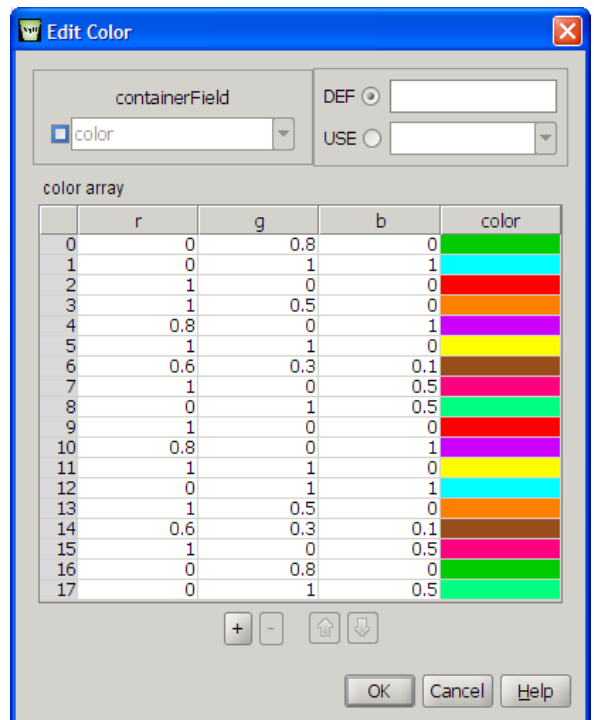
The value of the *colorPerVertex* field is ignored and always treated as true.






<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/IndexedTriangleSet.x3d>

*X3D for Web Authors*, Figure 13.8, p. 373. IndexedTriangleSet example using the same nine vertices previously defined.



 <b>IndexedTriangleSet</b>	<b>IndexedTriangleSet</b> is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node. <b>Hint:</b> insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
<b>DEF</b>	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. <b>Hint:</b> descriptive DEF names improve clarity and help document a model.
<b>USE</b>	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. <b>Hint:</b> USEing other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
<b>index</b>	<b>[index: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</b> (-1..+infinity) index specifies triangles by connecting Coordinate vertices.
<b>ccw</b>	<b>[ccw: accessType initializeOnly, type SBool (true/false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.
<b>colorPerVertex</b>	<b>[colorPerVertex: accessType initializeOnly, type SBool (true/false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
<b>normalPerVertex</b>	<b>[normalPerVertex: accessType initializeOnly, type SBool (true/false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
<b>solid</b>	<b>[solid: accessType initializeOnly, type SBool (true/false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
<b>containerField</b>	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
<b>class</b>	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#IndexedTriangleSet>

# IndexedTriangleFanSet node

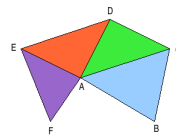
Indexed version of TriangleFanSet node

- Just fans and more fans

*index* array values define each individual fan, separated by -1 sentinel values

- First value in each *index* subsequence is fan center
- Efficient: each x-y-z point only needs to be defined once, since indexing can reuse it at any time

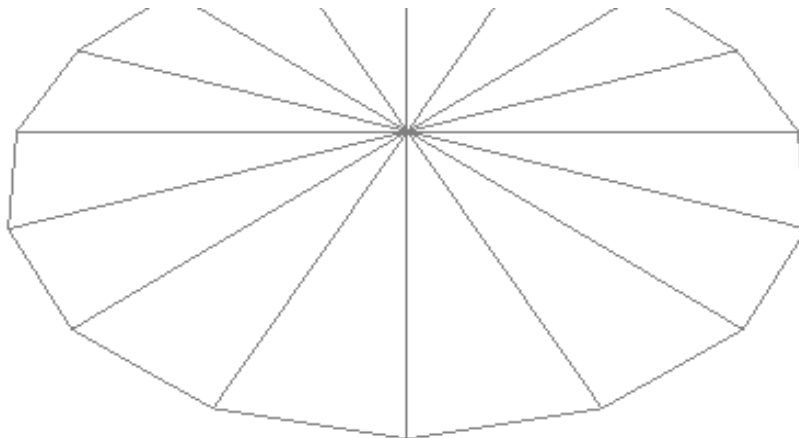
web|3D  
CONSORTIUM



The value of the *colorPerVertex* field is ignored and always treated as true.

Any convex polygon may be triangulated (i.e. tessellated) into a single fan by selecting one vertex as the fan center.

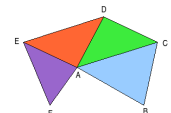
Example fan: look at how one browser tessellates a Cylinder end cap, displayed below in wireframe mode.

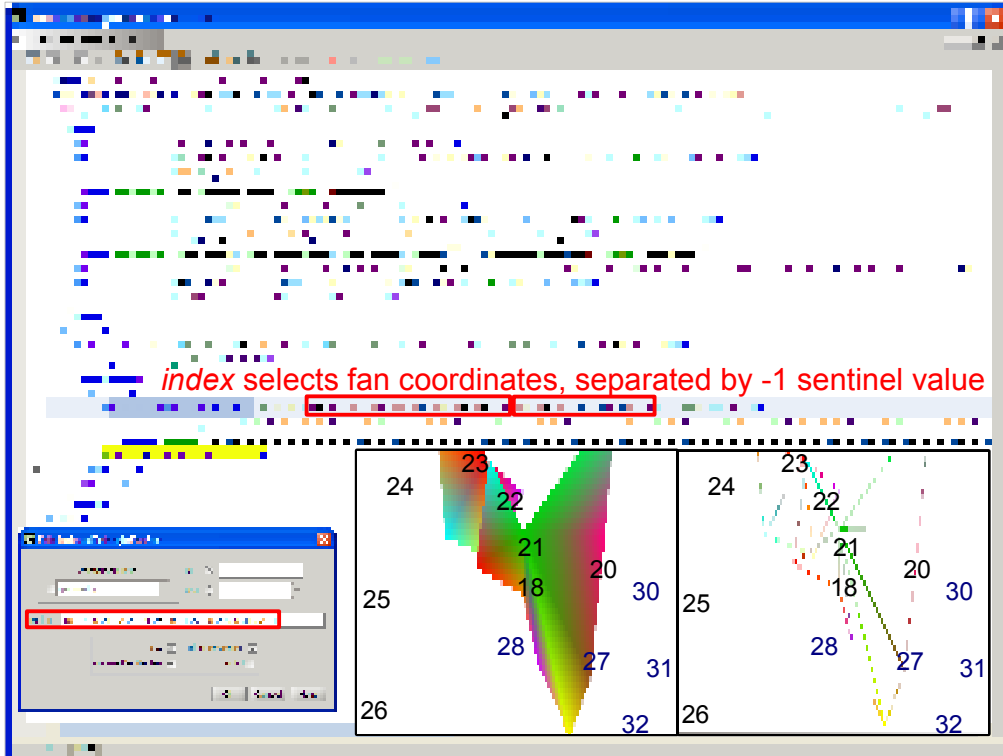


Hand-held fan image published with permission from  
[http://en.wikipedia.org/wiki/Image:Non\\_electric\\_fan\\_aka\\_sol-fjader.jpg](http://en.wikipedia.org/wiki/Image:Non_electric_fan_aka_sol-fjader.jpg)



Triangle fan image published with permission from  
[http://en.wikipedia.org/wiki/Triangle\\_fan](http://en.wikipedia.org/wiki/Triangle_fan)






<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/IndexedTriangleFanSet.x3d>

*X3D for Web Authors*, Figure 13.9, p. 375. IndexedTriangleFanSet example showing two fan sets, effectively producing the same fan sets as the TriangleFanSet example.

Note in the polygon images above that vertex 19 (cyan) is not labeled since it is quite close to vertex 27.

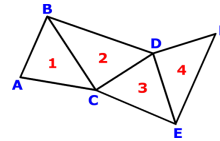
 IndexedTriangleFanSet	IndexedTriangleFanSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node. Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
DEF	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring all other attributes and children. Hint: USING other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
index	<b>[index: accessType initializeOnly, type MFloat32 CDATA #IMPLIED]</b> (-1..+infinity) index specifies triangles by connecting Coordinate vertices.
ccw	<b>[ccw: accessType initializeOnly, type SBoolean (true/false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.
colorPerVertex	<b>[colorPerVertex: accessType initializeOnly, type SBoolean (true/false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
normalPerVertex	<b>[normalPerVertex: accessType initializeOnly, type SBoolean (true/false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
solid	<b>[solid: accessType initializeOnly, type SBoolean (true/false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
containerField	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#IndexedTriangleFanSet>

# IndexedTriangleStripSet node

Indexed version of TriangleStripSet node

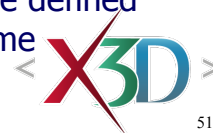
- Just a series of strip sets



*index* array values define each individual strip, separated by -1 sentinel values

- First three points referenced by *index* subsequence define initial triangle for strip
- Each subsequent index value adds another point, which in turn defines another triangle
- Efficient: each x-y-z point only needs to be defined once, since indexing can reuse it at any time

web|3D  
CONSORTIUM



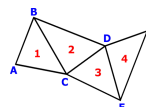
Large complex geometry may be split into several strip sets. Since Coordinate nodes can be repeatedly referenced via DEF/USE, common vertices shared by multiple strips do not need to be repeated. Instead, a single comprehensive Coordinate node can contain all of the necessary vertices for a complex piece of geometry, without duplications. Each IndexedTriangleStripSet re-USE-ing the comprehensive Coordinate node then provides the corresponding indexing for each individual strip.

Similarly, consistent coloring and normal values can be provided by Color, Normal and TextureCoordinate nodes that match the comprehensive Coordinate node. This approach can thus reduce file size by eliminating duplicate values. Some filesize-reduction tools (such as [Chisel](#)) provide such capabilities.

TODO: does the new Xj3D CadFilter capability support such simplification?

The value of the *colorPerVertex* field is ignored and always treated as true.


Triangle strip image published with permission from [http://en.wikipedia.org/wiki/Triangle\\_strip](http://en.wikipedia.org/wiki/Triangle_strip)





<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/IndexedTriangleStripSet.x3d>

*X3D for Web Authors*, Figure 13.10, p. 377. IndexedTriangleStripSet example, producing two strips using the same nine vertices as in many of the other examples. These strips match the previous TriangleStripSet example.

	<b>IndexedTriangleStripSet</b> is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node. <b>Hint:</b> insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
<b>DEF</b>	<b>[DEF ID #IMPLIED]</b> DEF defines a unique ID name for this node, referenceable by other nodes. <b>Hint:</b> descriptive DEF names improve clarity and help document a model.
<b>USE</b>	<b>[USE IDREF #IMPLIED]</b> USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. <b>Hint:</b> USING other geometry (instead of duplicating nodes) can improve performance. <b>Warning:</b> do NOT include DEF (or any other attribute values) when using a USE attribute!
<b>index</b>	<b>[index: accessType initializeOnly, type MInt32 CDATA #IMPLIED]</b> (-1..+infinity) index specifies triangles by connecting Coordinate vertices.
<b>ccw</b>	<b>[ccw: accessType initializeOnly, type SBool (true/false) "true"]</b> ccw = counterclockwise: ordering of vertex coordinates orientation. <b>Hint:</b> ccw false can reverse solid (backface culling) and normal-vector orientation.
<b>colorPerVertex</b>	<b>[colorPerVertex: accessType initializeOnly, type SBool (true/false) "true"]</b> Whether Color node is applied per vertex (true) or per polygon (false).
<b>normalPerVertex</b>	<b>[normalPerVertex: accessType initializeOnly, type SBool (true/false) "true"]</b> Whether Normal node is applied per vertex (true) or per polygon (false).
<b>solid</b>	<b>[solid: accessType initializeOnly, type SBool (true/false) "true"]</b> Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). <b>Warning:</b> default value true can completely hide geometry if viewed from wrong side!
<b>containerField</b>	<b>[containerField: NMTOKEN "geometry"]</b> containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
<b>class</b>	<b>[class CDATA #IMPLIED]</b> class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#IndexedTriangleStripSet>



## IndexedQuadSet node

Defines a set of quadrilaterals (rectangles)

- Coordinate *point* value sequences need to be planar

Sequential 4-tuples of *index* array values define quads, so no -1 sentinel values are inserted

- Each a-b-c-d set of *point* quadruplets in Coordinate node defines corners of an independent quad
- Efficient: each x-y-z point only needs to be defined once, since indexing can reuse it at any time

Hint: must set CADGeometry component level 1

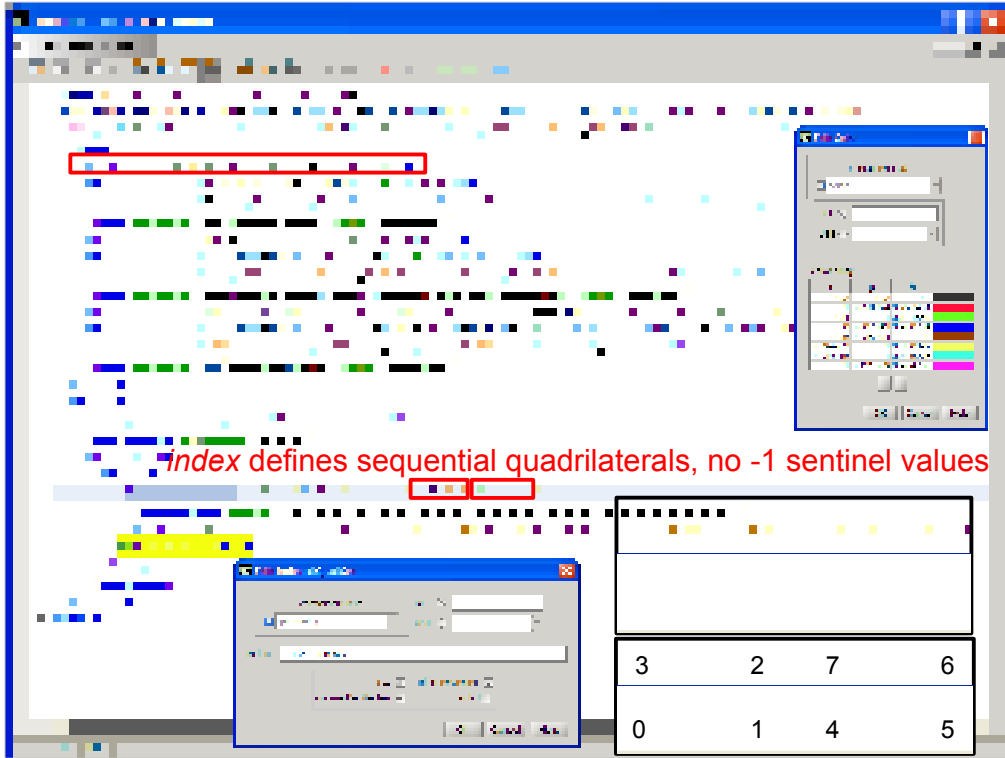


The value of *colorPerVertex* is always ignored and treated as false.


IndexedQuadSet does not include the *convex* field. Software renderers (or perhaps more accurately, the underlying graphics hardware cards) are expected to be able to handle concave quadrilaterals satisfactorily, since concavity of a 4-sided polygon is easier to compute than a concave n-sided polygon.

Any quadrilateral can be split into a pair of triangles. Two different splits are possible, matching the two internal diagonals. This is illustrated on the [Nonplanar quadrilaterals](#) slide.

The IndexedQuadSet is somewhat similar to the Rectangle2D node, defined in Chapter 10 Geometry2D. The primary difference is that IndexedQuadSet can be out of the X-Y plane and is defined using 3D coordinates.



<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter13-GeometryTrianglesQuadrilaterals/IndexedQuadSet.x3d>

 IndexedQuadSet	[X3D 3.1] IndexedQuadSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node. Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. Hint: USING other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
index	[index: accessType initializeOnly, type MInt32 CDATA #IMPLIED] coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets. Use -1 to separate indices for each polygon.
ccw	[ccw: accessType initializeOnly, type SBool (true/false) "true"] ccw = counterclockwise: ordering of vertex coordinates orientation. Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.
colorPerVertex	[colorPerVertex: accessType initializeOnly, type SBool (true/false) "true"] Whether Color node is applied per vertex (true) or per polygon (false).
normalPerVertex	[normalPerVertex: accessType initializeOnly, type SBool (true/false) "true"] Whether Normal node is applied per vertex (true) or per polygon (false).
solid	[solid: accessType initializeOnly, type SBool (true/false) "true"] Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). Warning: default value true can completely hide geometry if viewed from wrong side!
containerField	[containerField: NMTOKEN "geometry"] containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	[class CDATA #IMPLIED] class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#IndexedQuadSet>

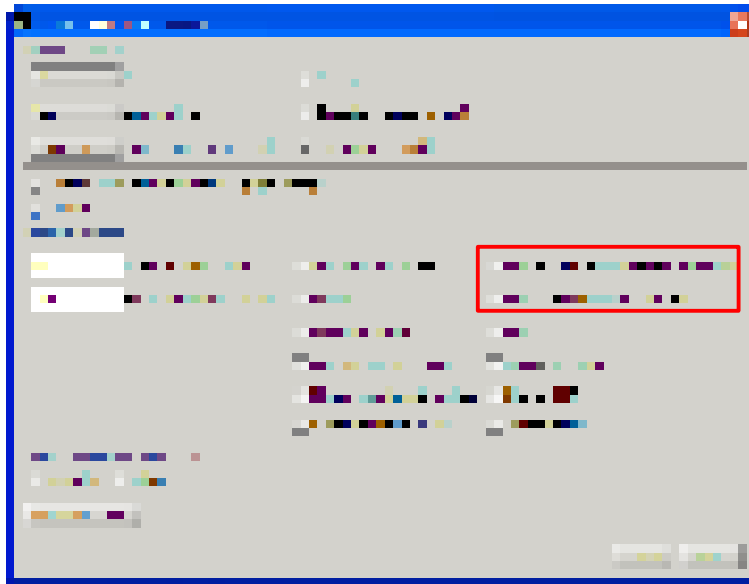
[back to Table of Contents](#)

## Additional Resources



57

## Xj3D CAD Filter Converter



Available in X3D-Edit:

- X3D > Tools > Conversion submenu
- This interface is selected within .x3d file editor pane via right-click context submenu

# Xj3D converterHelp.txt

Running Xj3D Format Converter converter.bat

CDFFilter - usage: filter [filters] input output [-loglevel type]  
[-exportVersion n] [-compressionMethod n ] [-quantization n ] [-upgrade] [filter\_args]

-loglevel type [ALL|WARNINGS|ERRORS|FATAL|NONE]

The minimum level that logs should be written at

-exportVersion n.n

The exported version of the X3D specification to generate  
No error checking is performed for invalid version numbers  
Assumes 3.1 if not supplied

-compressionMethod [FASTEST|SMALLEST|LOSSY|STRINGS]

Define what sort of compression algorithms should be used  
when X3D Binary format is used for the output. Ignored in  
all other cases

-quantization n

Positive floating point value that states how much quantization  
of values is allowed. Default is 0.001

-upgrade

When declared, any VRML style PROTO content that can be  
upgraded to X3D native nodes, will be

Available filters:

- MinProfile
- CombineShapes
- Triangulation
- TriangleCountInfo
- GenNormals
- ShortenDEF
- IFSToTS
- ReIndex
- Debug
- DEFUSEImageTexture
- AbsScale
- Index
- Identity
- ModifyViewpoint
- Center
- FlattenTransform
- IFSToITS

CAD Filter Converter is also available via command line for locally installed Xj3D.

On Windows systems Xj3d is typically installed at C:\Program Files\Xj3D

# Chisel

## Chisel VRML Optimisation Tool

- Features: Validate, Format, Clean, Condense, Reduce, Reorganize, Mutate
- Autoinstaller <http://www2.hrp.no/vr/tools/chisel/install.htm>
- Documentation <http://www2.hrp.no/vr/tools/chisel/doc>

## Open source Java

- Supported by Halden Virtual Reality Centre, Norway
- Originally built by Trapezium, maintained by NIST

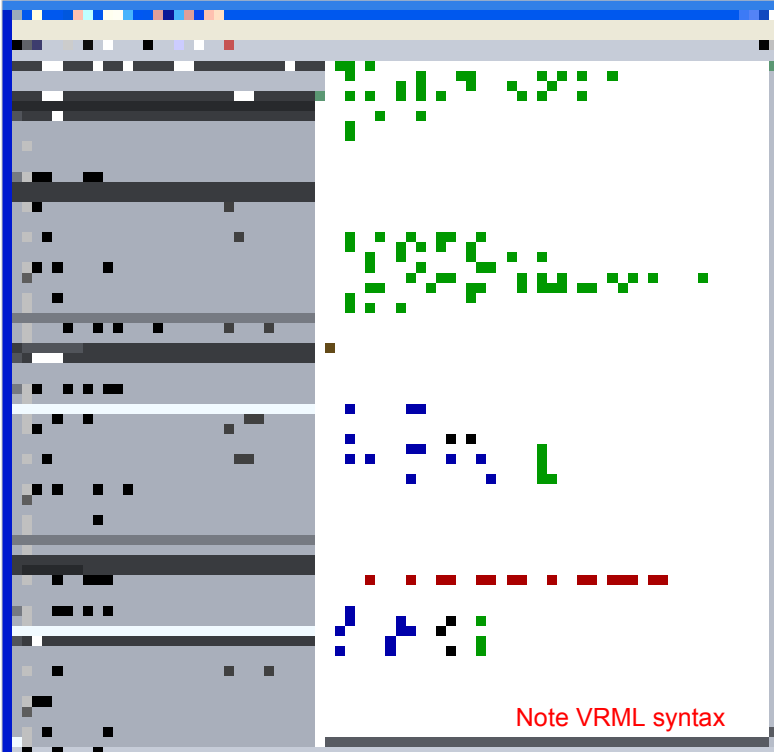
Usage: export to VRML, modify, import back to X3D

Wish list: .x3d handling, X3D-Edit integration



Other conversion tools (including polygon decimators, which combine nearly coincident/coplanar triangles) are listed on X3D Resources page under Conversions and Translation Tools.

<http://www.web3d.org/x3d/content/examples/X3dResources.html#Conversions>



Chisel Features

- fine-tune control of changes
- statistics for size, polygons
- geometric compression
- gzip compression

< ~~X~~3D >

61

Note VRML syntax



# OpenGL Programming API

OpenGL home has a vast set of specifications and programming resources for the underlying graphics rendering software

- X3D players typically use OpenGL or DirectX to drive the underlying graphics hardware
- With effort, programming tools and algorithms can sometimes be adapted to work with X3D constructs
- <http://www.opengl.org>

OpenGL Programming Guide

- <http://www.glprogramming.com/red>



OpenGL programming is much different than X3D scene graph authoring. Most of the resources on these sites are probably only appropriate for experienced programmers.

Java programmers might want to look at JOGL, the Java OpenGL applications programming interface (API). <https://jogl.dev.java.net>

Note that none of the OpenGL API codebases are directly suitable for use in the Script node as part of an X3D scene. However the algorithms can often be rewritten for the X3D Scene Access Interface (SAI), described in Chapter 9 Scripting.

[back to Table of Contents](#)

## Chapter Summary



63

## Summary: Triangles and Quadrilaterals

### Common fields

- *solid, ccw, colorPerVertex, normalPerVertex*

Review from  
Chapters 2,6

### Normal vectors, Normal node

### Ordered polygonal nodes

- TriangleSet, TriangleFanSet, TriangleStripSet, and QuadSet

### Indexed polygonal nodes

- IndexedTriangleSet, IndexedTriangleFanSet, IndexedTriangleStripSet, and IndexedQuadSet

Pay close attention to detail with these nodes!

## Suggested exercises

Build a Box or a Platonic Solid out of triangles

Write small program to compute polygon values

- Text output of program: viewable .x3d scene file

Apply an image or movie to a billboarded Quad

- Example: Chapter 5, MovieTextureAuthoringOptions.x3d

Convert a simple IndexedFaceSet node into each of TriangleSet nodes to compare differences

- Using pencil + graph paper will help

Test Xj3D CAD Filter Converter using X3D-Edit

- IndexedFaceSet to TriangleSet or IndexedTriangleSet

[back to Table of Contents](#)

## References



66

# References 1

*X3D: Extensible 3D Graphics for Web Authors*  
by Don Brutzman and Leonard Daly, Morgan  
Kaufmann Publishers, April 2007, 468 pages.



- Chapter 13, Geometry Nodes part 4:  
Triangles and Quadrilaterals
- <http://x3dGraphics.com>
- <http://x3dgraphics.com/examples/X3dForWebAuthors>

## X3D Resources

- <http://www.web3d.org/x3d/content/examples/X3dResources.html>



67

## References 2

### X3D-Edit Authoring Tool

- <https://savage.nps.edu/X3D-Edit>

### X3D Scene Authoring Hints

- <http://x3dgraphics.com/examples/X3dSceneAuthoringHints.html>

### X3D Graphics Specification

- <http://www.web3d.org/x3d/specifications>
- Also available as help pages within X3D-Edit



68

## References 3

*VRML 2.0 Sourcebook* by Andrea L. Ames, David R. Nadeau, and John L. Moreland, John Wiley & Sons, 1996.



- <http://www.wiley.com/legacy/compbooks/vrml2sbk/cover/cover.htm>
- <http://www.web3d.org/x3d/content/examples/Vrml2.0Sourcebook>
- Chapter 13 - Points Lines Faces covers common fields and related IndexedFaceSet examples

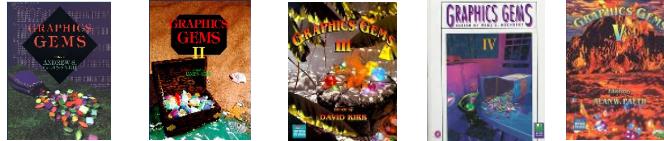
Note: X3D triangle and quadrilateral nodes in this chapter were not yet defined in VRML97



## References 4

*Graphics Gems* book series: many algorithms

- <http://www.graphicsgems.org>



Journal of Graphics Tools (jgt): many algorithms

- <http://jgt.akpeters.com>



web|3D  
CONSORTIUM



<http://www.merriam-webster.com/dictionary/algorithm>

Term: al·go·rithm

Pronunciation: \ 'al-gə-,ri-thəm \

Function: noun

Etymology: alteration of Middle English algorisme, from Old French & Medieval Latin; Old French, from Medieval Latin algorismus, from Arabic al-khuwārizmi, from al-Khwārizmī fl a.d. 825 Islamic mathematician

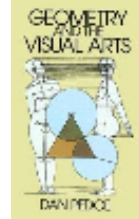
Date: 1926

Definition: a procedure for solving a mathematical problem (as of finding the greatest common divisor) in a finite number of steps that frequently involves repetition of an operation; broadly : a step-by-step procedure for solving a problem or accomplishing some end especially by a computer

Important requirement: a proper algorithm includes a termination condition.

## References 5

*Geometry and the Visual Arts*, Daniel Pedoe, Dover Publications, 1983.



### Wikipedia

- Triangle strip
- Triangle fan
- Platonic solids
- Archimedean solids

web|3D  
CONSORTIUM



71

### Virtual Polyhedra

- The Encyclopedia of Polyhedra by George W. Hart
- Interesting site, but models are VRML 1.0 which requires special conversion
- <http://www.georgehart.com/virtual-polyhedra/vp.html>

### Interactive polyhedra by Evgeny Demidov

- <http://ibiblio.org/e-notes/3Dapp/Convex.htm>

# Contact

**Don Brutzman**

*[brutzman@nps.edu](mailto:brutzman@nps.edu)*

*<http://faculty.nps.edu/brutzman>*

Code USW/Br, Naval Postgraduate School  
Monterey California 93943-5000 USA  
1.831.656.2149 voice

web|**3D**  
CONSORTIUM



# CGEMS, SIGGRAPH, Eurographics

The Computer Graphics Educational Materials Source(CGEMS) site is designed for educators

- to provide a source of refereed high-quality content
- as a service to the Computer Graphics community
- freely available, directly prepared for classroom use
- <http://cgems.inesc.pt>

*X3D for Web Authors* recognized by CGEMS! ☺

- Book materials: X3D-Edit tool, examples, slidesets
- Received jury award for Best Submission 2008

CGEMS supported by SIGGRAPH, Eurographics



From the CGEMS home page:

- <http://cgems.inesc.pt>

Welcome to CGEMS - Computer Graphics Educational Materials Source. The CGEMS site is designed for educators to provide a source of refereed high-quality content as a service to the Computer Graphics community as a whole. Materials herein are freely available and directly prepared for your classroom.

List of all published modules:

- <http://cgems.inesc.pt/authors/ListModules.aspx>

CGEMS Editorial Policy:

- <http://cgems.inesc.pt/EditorialPolicy.htm>

## Creative Commons open-source license

<http://creativecommons.org/licenses/by-nc-sa/3.0>



### Attribution-Noncommercial-Share Alike 3.0 Unported

You are free:

- \* to Share — to copy, distribute and transmit the work
- \* to Remix — to adapt the work

Under the following conditions:

\* Attribution. You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).

Attribute this work: What does "Attribute this work" mean?

The page you came from contained embedded licensing metadata, including how the creator wishes to be attributed for re-use. You can use the HTML here to cite the work. Doing so will also include metadata on your page so that others can find the original work as well.

- \* Noncommercial. You may not use this work for commercial purposes.
- \* Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.
- \* For any reuse or distribution, you must make clear to others the license terms of this work. The best way to do this is with a link to this web page.
- \* Any of the above conditions can be waived if you get permission from the copyright holder.
- \* Nothing in this license impairs or restricts the author's moral rights.

# Open-source license for X3D-Edit software and X3D example scenes

<http://www.web3d.org/x3d/content/examples/license.html>

Copyright (c) 1995-2013 held by the author(s). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the names of the Naval Postgraduate School (NPS) Modeling Virtual Environments and Simulation (MOVES) Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

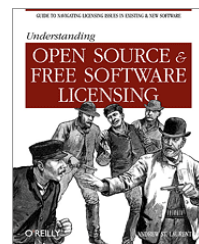
License available at

<http://www.web3d.org/x3d/content/examples/license.txt>

<http://www.web3d.org/x3d/content/examples/license.html>

Good references on open source:

Andrew M. St. Laurent, *Understanding Open Source and Free Software Licensing*, O'Reilly Publishing, Sebastopol California, August 2004. <http://oreilly.com/catalog/9780596005818/index.html>



Herz, J. C., Mark Lucas, John Scott, *Open Technology Development: Roadmap Plan*, Deputy Under Secretary of Defense for Advanced Systems and Concepts, Washington DC, April 2006. <http://handle.dtic.mil/100.2/ADA450769>

