

X3D Graphics for Web Authors

Chapter 5

Appearance, Material, and Textures

Things are not always as they appear.

Contents

Chapter Overview and Concepts

X3D Nodes and Examples

Additional Resources

Chapter Summary

References

Chapter Overview

Overview: Appearance, Material, and Textures

Appearance affects associated geometry,
containing the following fields

Visual surface properties that interact with lights

- Material and TwoSidedMaterial
- LineProperties and FillProperties

Texture nodes wrap images onto geometry

- ImageTexture, MovieTexture, PixelTexture and MultiTexture
- TextureTransform, TextureCoordinate and TextureCoordinateGenerator

Concepts

Motivation

Appearance, material and texture nodes are intended to allow authors to make 3D objects look similar to objects in the real world

- This goal is always a worthy challenge

Lighting is an important factor in appearance, because 3D objects reflect their virtual light

- Appearance and lighting are computational
- In this chapter we assume white light available, usually from default NavigationInfo headlight
- Lighting and environment covered in Chapter 11

Parent-child constraints

Each Shape node can contain

- Single geometry node
- Single Appearance node

Each Appearance node can contain

- Single Material (or TwoSidedMaterial) node
- FillProperties, LineProperties, single Texture node

Each Texture node can contain

- Single TextureTransform or
TextureTransformGenerator node

Common functionality

Node repetition can be efficiently accomplished via DEF and USE

- Remember, first DEF must precede any USE copies
- Simplifies application of consistent coloring to multiple pieces of geometry which are either similar or parts of the same larger object

Consistent, more efficient, easier to globally change all instances at once

- Which is further important when changing styles or applying accessibility techniques throughout

X3D Nodes and Examples

Appearance node

Each Shape contains some geometry along with a corresponding Appearance node

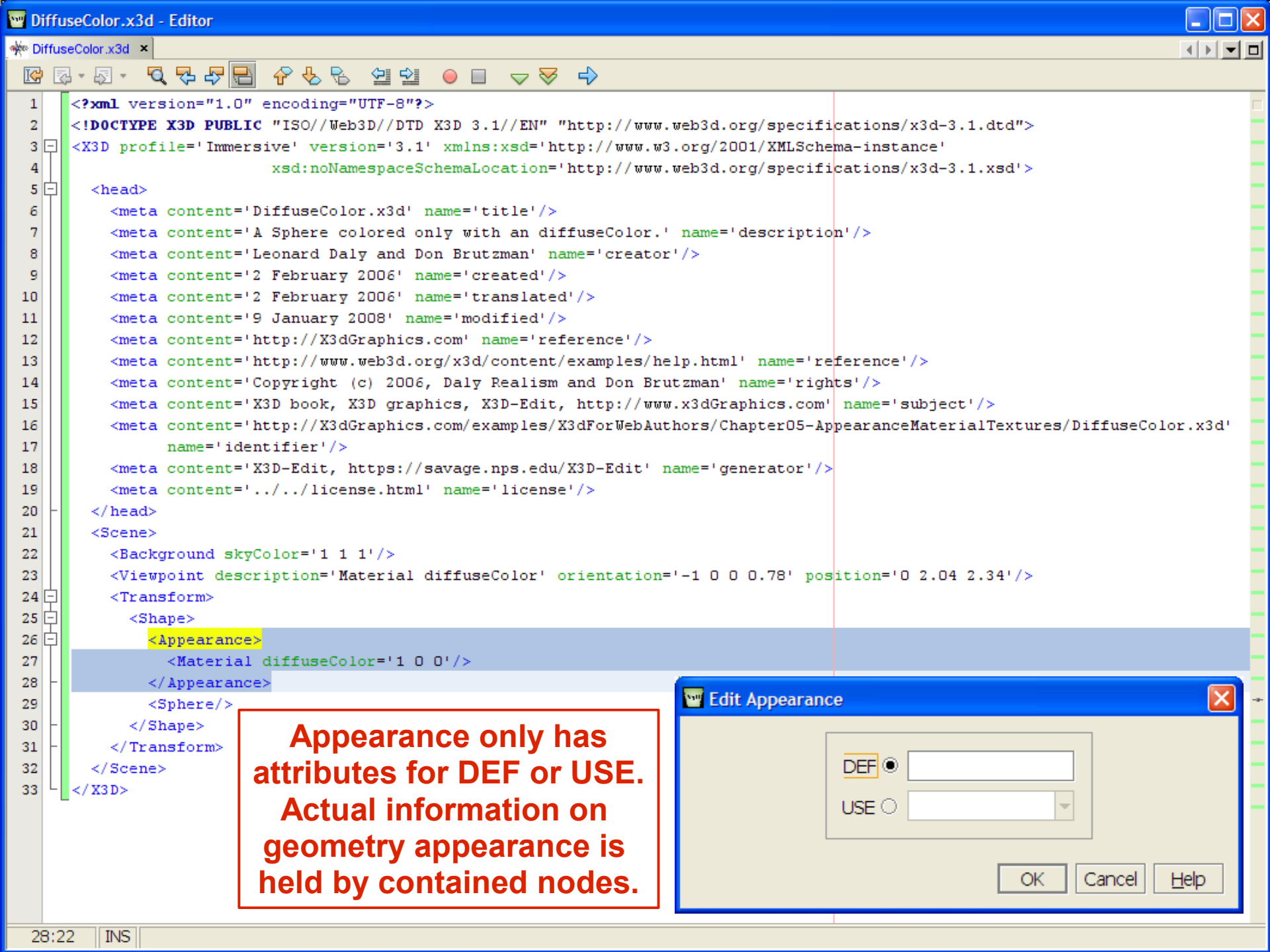
Appearance is a container which may include

- Single Material (or TwoSidedMaterial) node
- FillProperties, LineProperties, single Texture node

This close association makes assignment of rendering properties to geometry unambiguous

- Repetition of values for visual consistency is easily accomplished with DEF/USE of Appearance, Material, Texture node, etc.
- Clear naming helps, for example

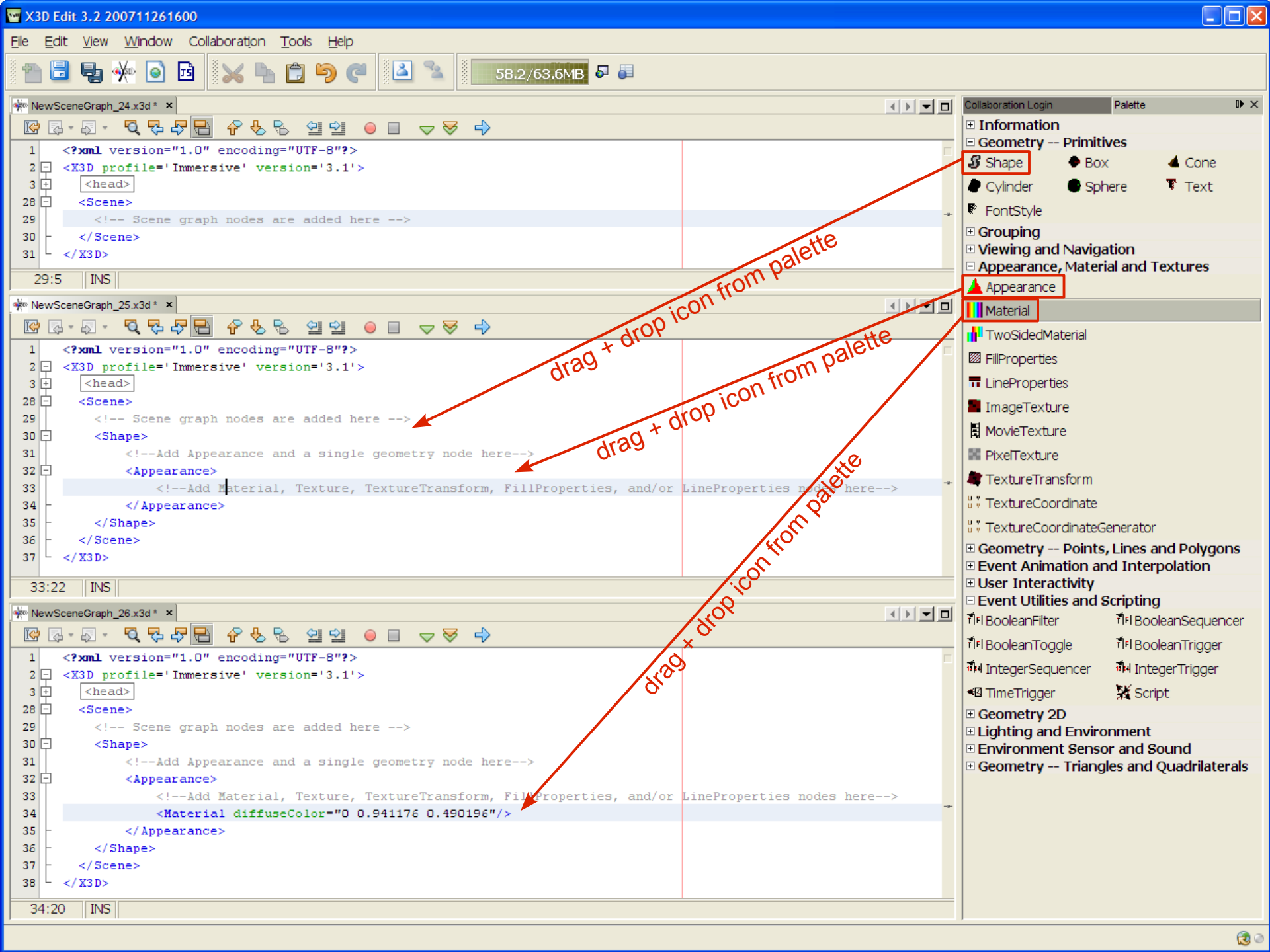
```
<Appearance USE='FoggyGlassAppearance' />
```




Palette simplifies addition of new nodes

Use X3D-Edit palette to pick the node of interest:

- Palette groups match chapter structure, and can be reordered by dragging with mouse
- Upon dragging a new node element into scene, corresponding node editor pops up
- After checking attribute values with node editor, select OK to confirm the new node
- Default attribute values omitted in XML for clarity
- Erroneous node placement in scene graph, or invalid attribute values, cause a validation error
- Accept or reject validation errors as appropriate, then continue with text editing if desired



 Appearance	Appearance specifies the visual properties of geometry by containing the Material, Texture and TextureTransform nodes. Hint: insert a Shape node before adding geometry or Appearance. Interchange profile hint: only Material and ImageTexture are allowed.
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 <u>all</u> other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
containerField	[containerField: NMTOKEN "appearance"] 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.

Material node

Material controls how most geometry is colored, whether it is transparent or glowing, etc.

Surface visual properties are applied equally across all polygons making up a shape

Material properties define how geometry visually interacts with light sources in the scene

- Lighting and Environment is covered in Chapter 11
- Rendering results also depend on view perspective

Material is an important node to master

TwoSidedMaterial node

TwoSidedMaterial fields are identical to Material, with the addition of the following new fields:

- *backAmbientIntensity*
- *backDiffuseColor*, *backEmissiveColor*, *backSpecularColor*
- *backShininess*
- *backTransparency*

The 'back' fields determine how the 'backsides' of polygons are drawn

- Such as insides of primitive geometry
- Corresponding geometry must have *solid*='false'

Reading X3D Specification node signatures

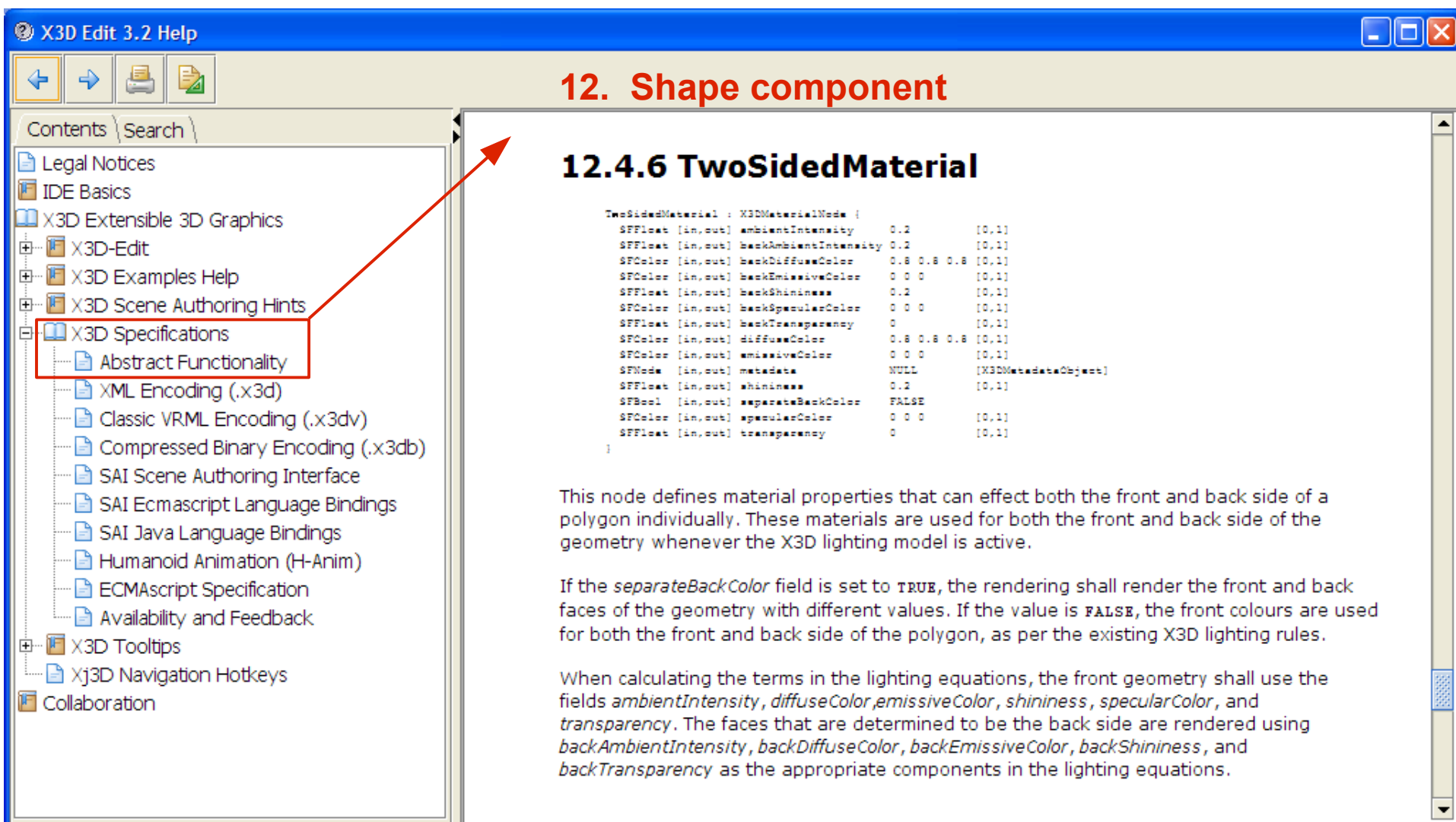
Actual X3D Specification entries are as follows:

- SFFloat [in,out] *backAmbientIntensity* 0.2 [0,1]
- SFCOLOR [in,out] *backDiffuseColor* 0.8 0.8 0.8 [0,1]
- SFCOLOR [in,out] *backEmissiveColor* 0 0 0 [0,1]
- SFFloat [in,out] *backShininess* 0.2 [0,1]
- SFCOLOR [in,out] *backSpecularColor* 0 0 0 [0,1]
- SFFloat [in,out] *backTransparency* 0 [0,1]

These field signatures are interpreted as follows:

- SFCOLOR and SFFloat are field types
- [in,out] is accessType
- default value is followed by [min,max]

TwoSidedMaterial specification help entry in X3D-Edit



The screenshot shows the X3D Edit 3.2 Help window. The left pane displays a table of contents with 'X3D Specifications' and 'Abstract Functionality' highlighted. A red arrow points from this entry to the main content area. The main content area is titled '12. Shape component' and '12.4.6 TwoSidedMaterial'. It contains a table of fields for the TwoSidedMaterial node and two paragraphs of text explaining its function.

12. Shape component

12.4.6 TwoSidedMaterial

Field	Type	Default	Range
ambientIntensity	SFFloat [in,out]	0.2	[0,1]
backAmbientIntensity	SFFloat [in,out]	0.2	[0,1]
backDiffuseColor	SFColor [in,out]	0.8 0.8 0.8	[0,1]
backEmissiveColor	SFColor [in,out]	0 0 0	[0,1]
backShininess	SFFloat [in,out]	0.2	[0,1]
backSpecularColor	SFColor [in,out]	0 0 0	[0,1]
backTransparency	SFFloat [in,out]	0	[0,1]
diffuseColor	SFColor [in,out]	0.8 0.8 0.8	[0,1]
emissiveColor	SFColor [in,out]	0 0 0	[0,1]
metadata	SFNode [in,out]	NULL	[X3DMetadataObject]
shininess	SFFloat [in,out]	0.2	[0,1]
separateBackColor	SFBool [in,out]	FALSE	
specularColor	SFColor [in,out]	0 0 0	[0,1]
transparency	SFFloat [in,out]	0	[0,1]

This node defines material properties that can effect both the front and back side of a polygon individually. These materials are used for both the front and back side of the geometry whenever the X3D lighting model is active.

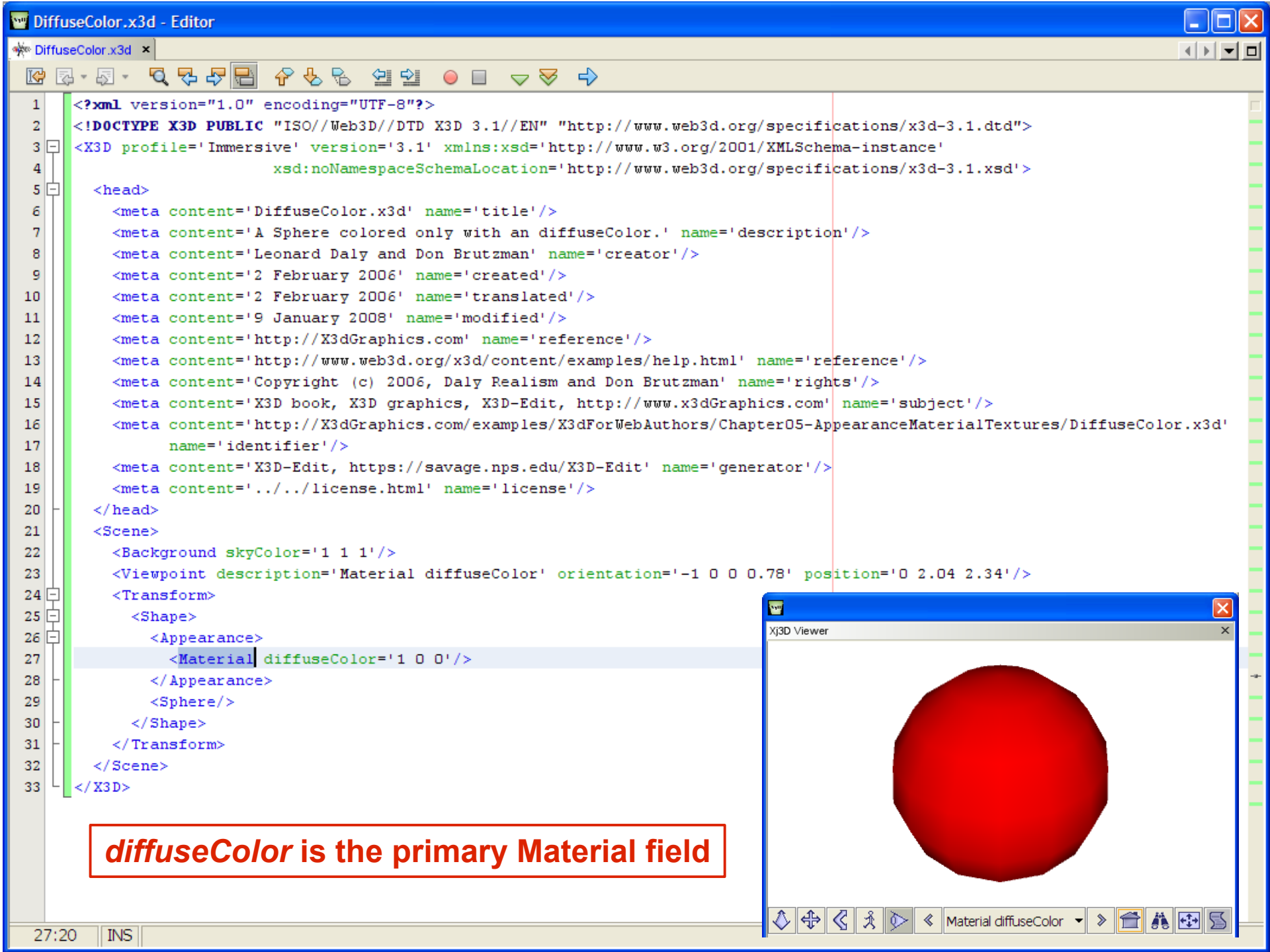
If the *separateBackColor* field is set to **TRUE**, the rendering shall render the front and back faces of the geometry with different values. If the value is **FALSE**, the front colours are used for both the front and back side of the polygon, as per the existing X3D lighting rules.

When calculating the terms in the lighting equations, the front geometry shall use the fields *ambientIntensity*, *diffuseColor*, *emissiveColor*, *shininess*, *specularColor*, and *transparency*. The faces that are determined to be the back side are rendered using *backAmbientIntensity*, *backDiffuseColor*, *backEmissiveColor*, *backShininess*, and *backTransparency* as the appropriate components in the lighting equations.

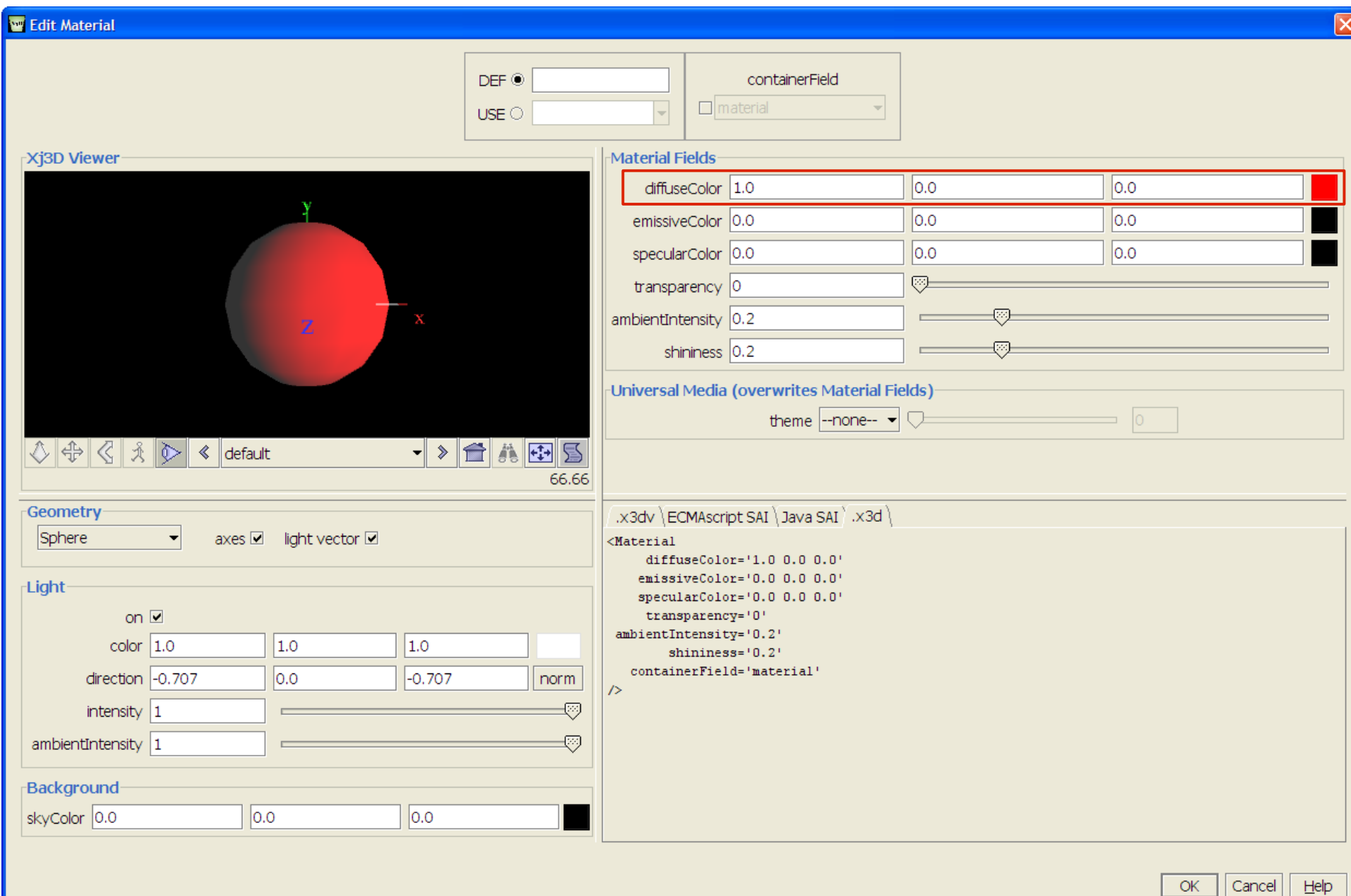
Material fields

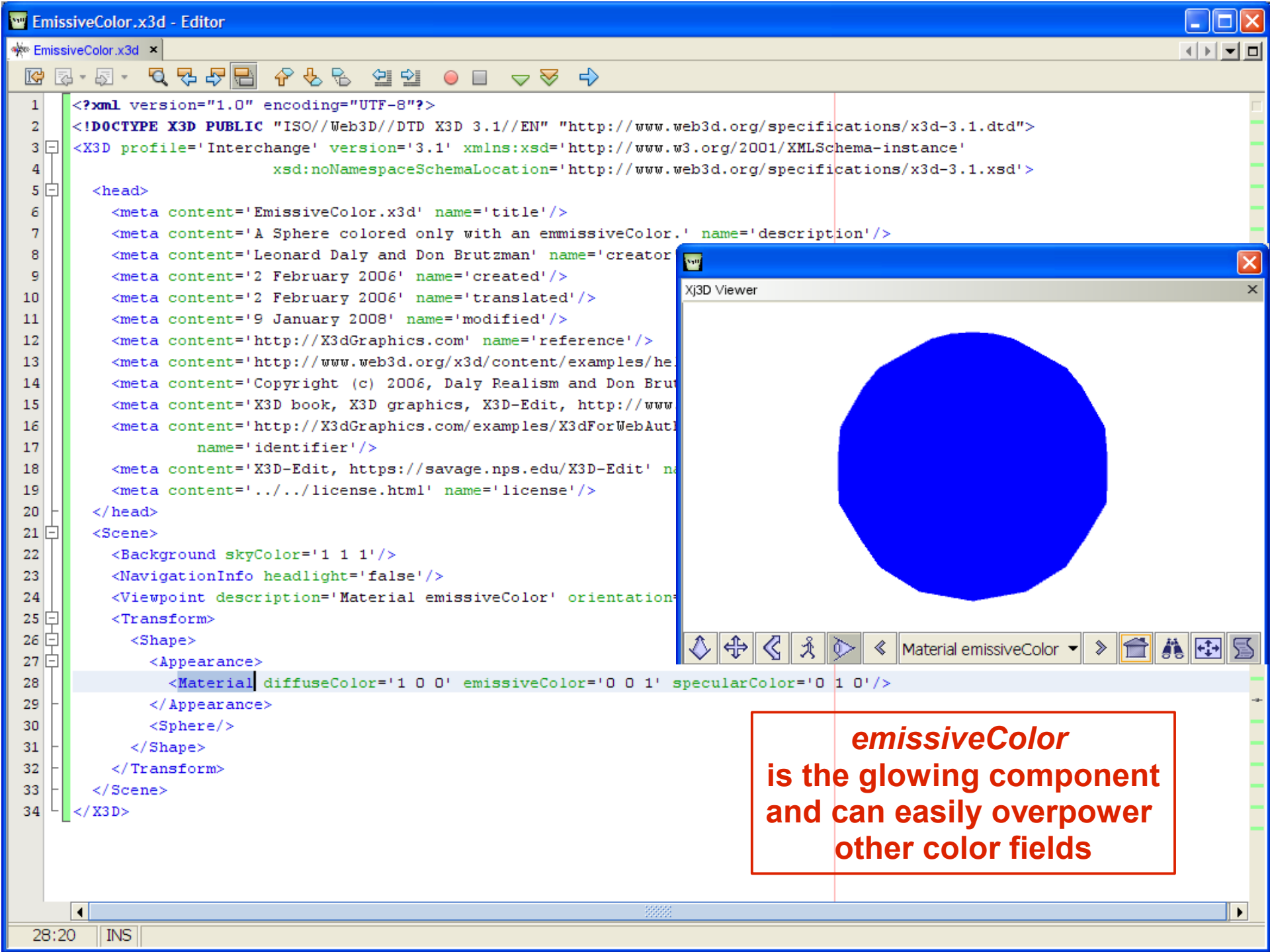
Color, transparency and shininess fields together make up Material properties. Examples follow.

- *diffuseColor* reflects all X3D light sources, depending on viewing angles towards each light
- *ambientIntensity* is reflection multiplication factor
- *emissiveColor* is glowing component, normally off, independent of reflected light
- *specularColor* governs reflection highlights
- *shininess* controls specular intensity
- *transparency* is ability to see through an object:
1 is completely transparent, 0 is opaque

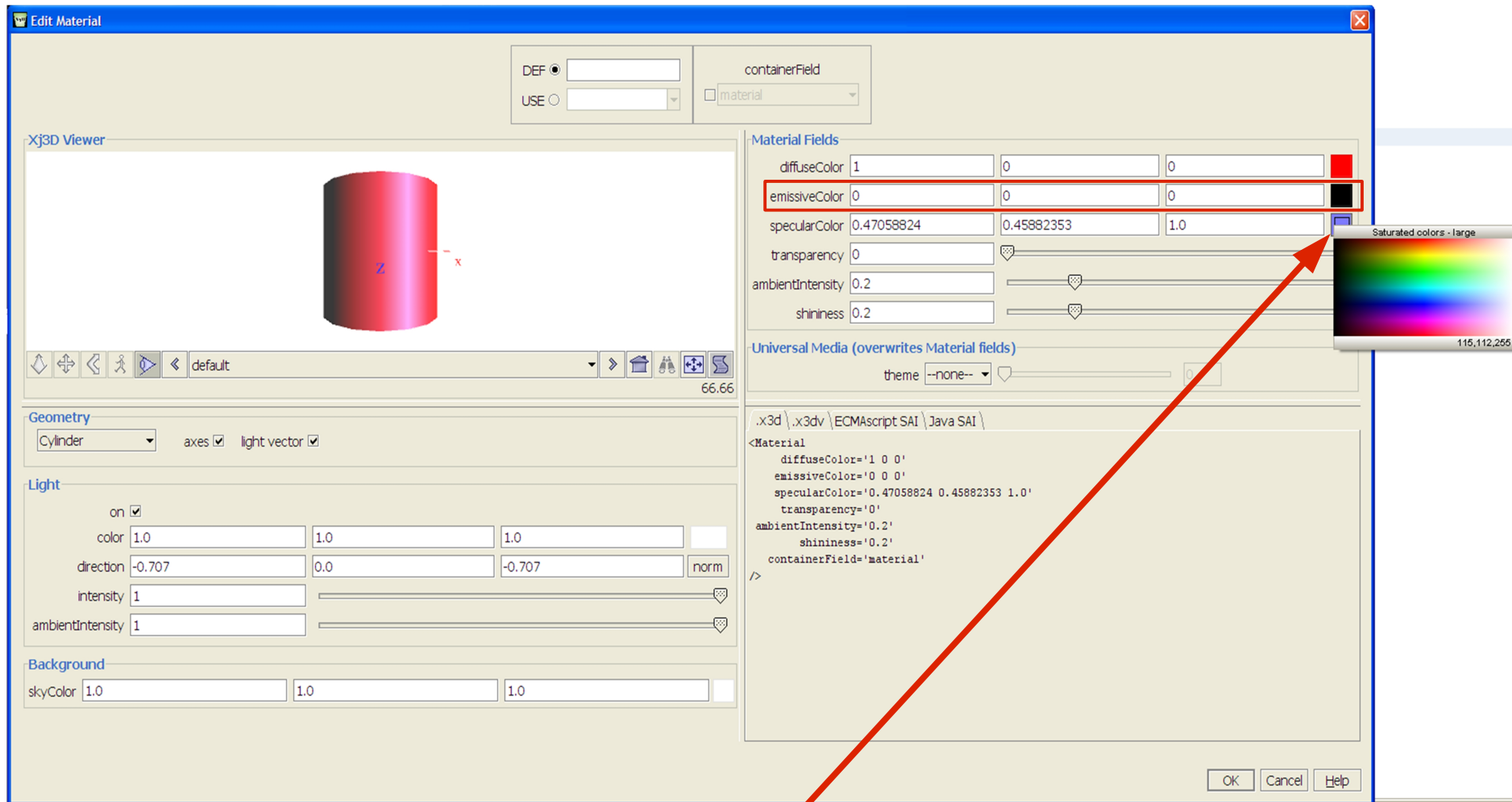


Material editor: *diffuseColor*

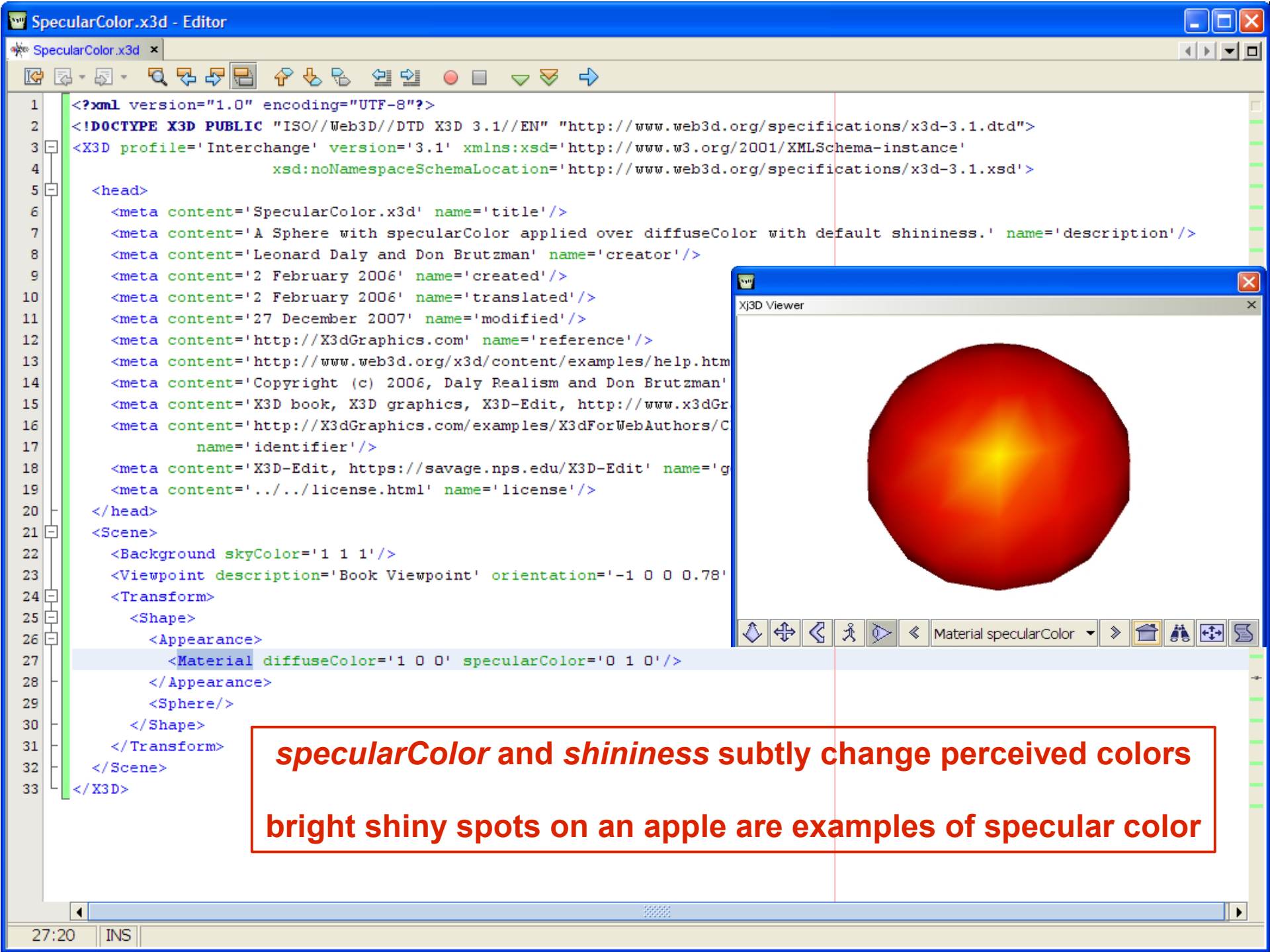




Material editor color selector

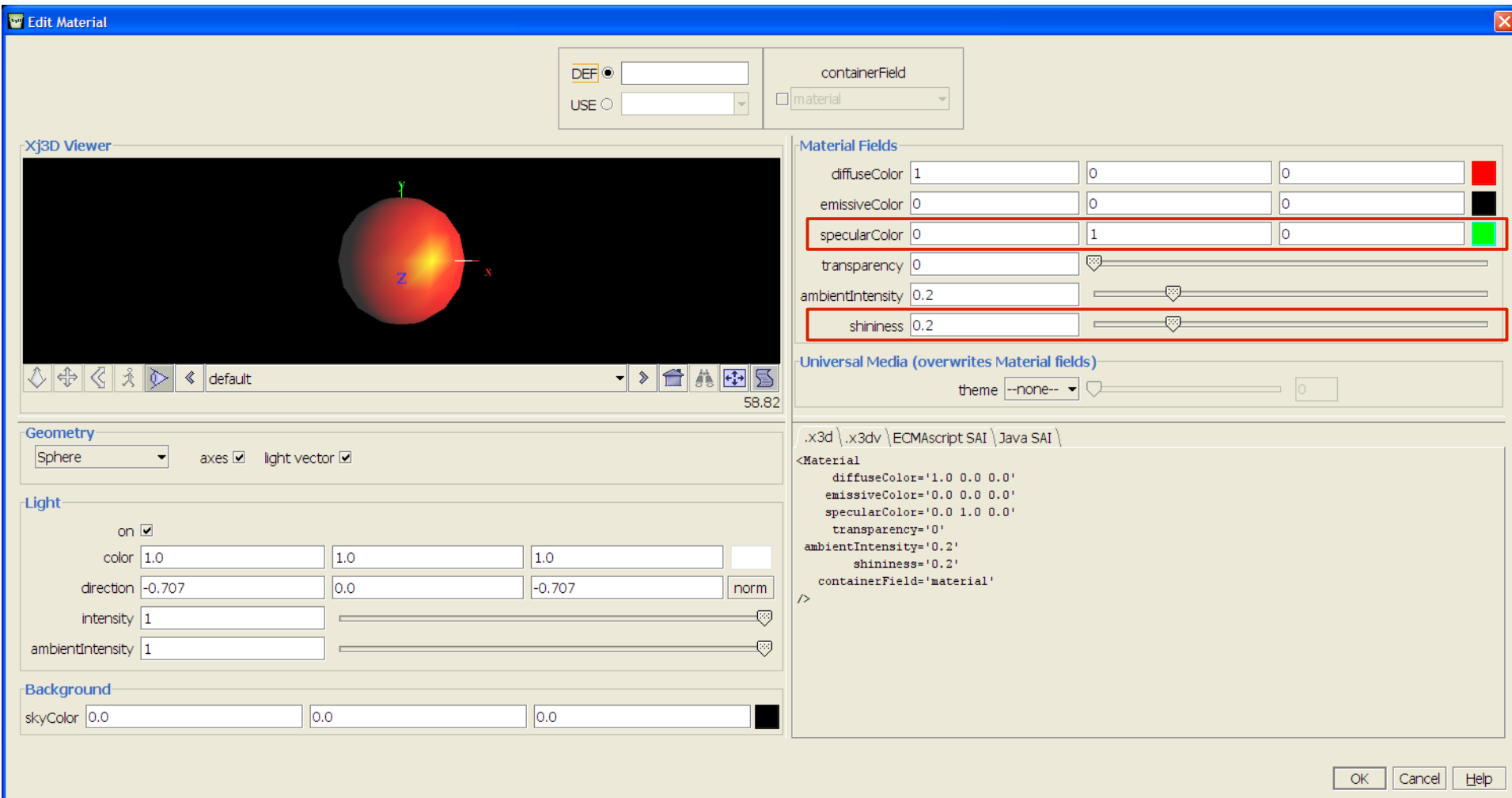


Click colored box to select a color



***specularColor* and *shininess* subtly change perceived colors**
bright shiny spots on an apple are examples of specular color

Material editor *specularColor, shininess*

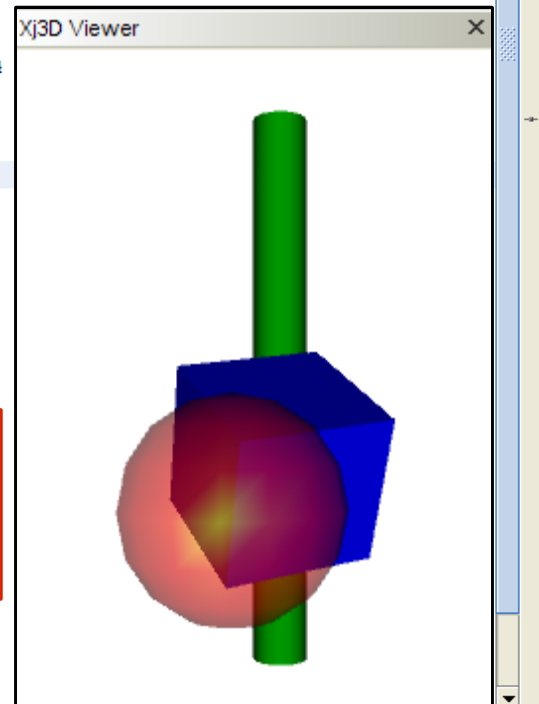


```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3 <X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/s
4 <head>
5   <meta content='Transparency.x3d' name='title' />
6   <meta content='This example shows a partially transparent Sphere in front of an opaque Box and Cylinder.' name='description' />
7   <meta content='Leonard Daly and Don Brutzman' name='creator' />
8   <meta content='2 February 2006' name='created' />
9   <meta content='2 February 2006' name='translated' />
10  <meta content='1 April 2007' name='modified' />
11  <meta content='http://X3dGraphics.com' name='reference' />
12  <meta content='http://www.web3d.org/x3d/content/examples/help.html' name='reference' />
13  <meta content='Copyright (c) 2006, Daly Realism and Don Brutzman' name='rights' />
14  <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dgraphics.com' name='subject' />
15  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter05-AppearanceMaterialTextures/Transparency.x3d' name='identifier' />
16  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
17  <meta content='../license.html' name='license' />
18 </head>
19 <Scene>
20   <ExternProtoDeclare name='WhereAmI' url='../Chapter14-Prototypes/WhereAmI.x3d#WhereAmI' "http://X3dGraphics.com/examples/X3dForWebAuthors/Chap
21   <ProtoInstance name='WhereAmI' />
22   <Background skyColor='1 1 1' />
23   <Viewport description='Book Viewpoint' orientation='-0.788 -0.614 -0.055 -0.12' position='-0.56 -0.64
24   <Transform>
25     <Shape>
26       <Appearance>
27         <Material diffuseColor='1 0 0' specularColor='0 1 0' transparency='.4' />
28       </Appearance>
29       <Sphere radius='.7' />
30     </Shape>
31   </Transform>
32   <Transform rotation='.707 .707 0 .707' translation='0 0 -6'>
33     <Shape>
34       <Appearance>
35         <Material diffuseColor='0 0 1' />
36       </Appearance>
37       <Box />
38     </Shape>
39   </Transform>
40   <Transform translation
41     <Shape>
42       <Appearance>
43         <Material diffuseColor='0 .6 0' />
44       </Appearance>
45       <Cylinder height='8' radius='.4' />
46     </Shape>

```

**View of semitransparent shiny Sphere,
in front of nontransparent (opaque) Box,
in front of opaque Cylinder**



Universal Media materials library

The Universal Media materials were originally created by SGI as part of OpenInventor in the 1990s as a convenience to authors

Each set of materials is grouped for visual compatibility and aesthetic appeal

Now converted and available for X3D use

- David Rousseau converted to VRML97
- Aaron Walsh created VRML Universal Media archive
- Don Brutzman translated into X3D as prototypes, cut/paste field values, also embedded in X3D-Edit
- <http://www.web3d.org/x3d/content/examples/Basic/UniversalMediaMaterials>

Selecting a Universal Material value

The screenshot shows the 'Edit Material' interface. The 'Xj3D Viewer' displays a red sphere. The 'Material Fields' section contains sliders for diffuseColor, emissiveColor, specularColor, transparency, ambientIntensity, and shininess. The 'Universal Media (overwrites Material fields)' section features a 'theme' dropdown menu with a list of materials: Metal, Neon, Rococo, SantaFe, Sheen, Silky, Spring, and Summer. The 'SantaFe' material is selected, and the index '9' is entered next to it. A red box highlights this selection, and a red arrow points from a text box to it.

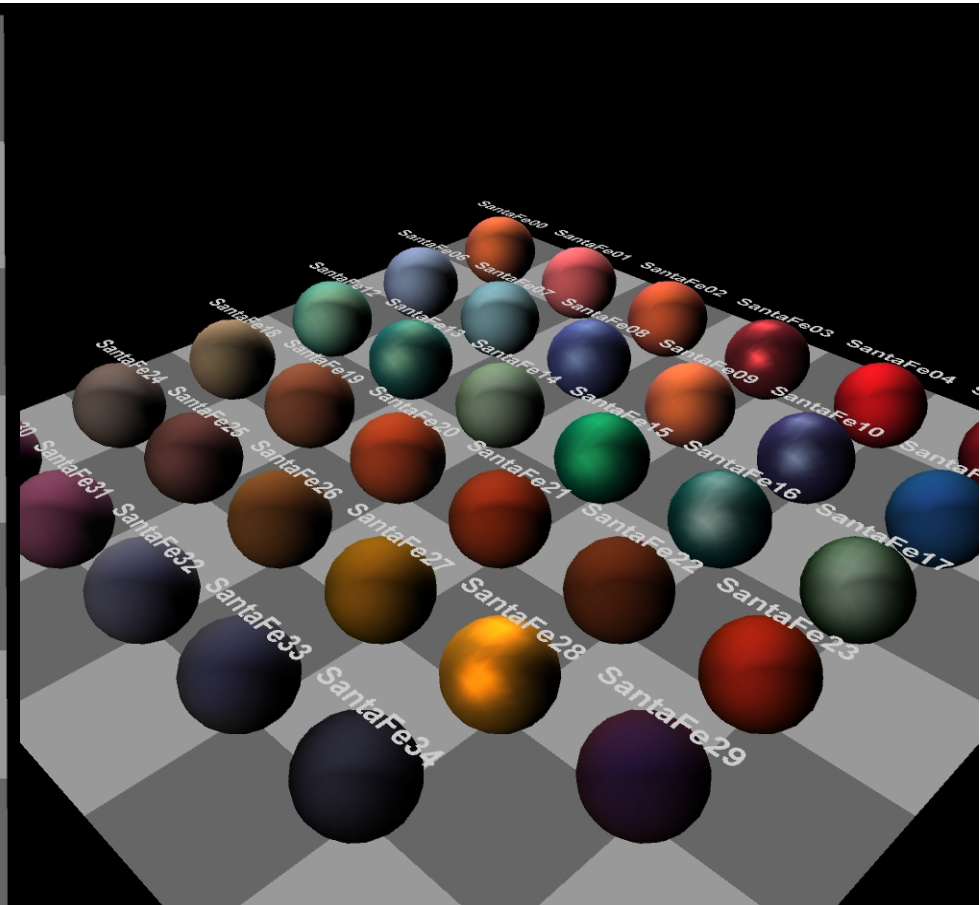
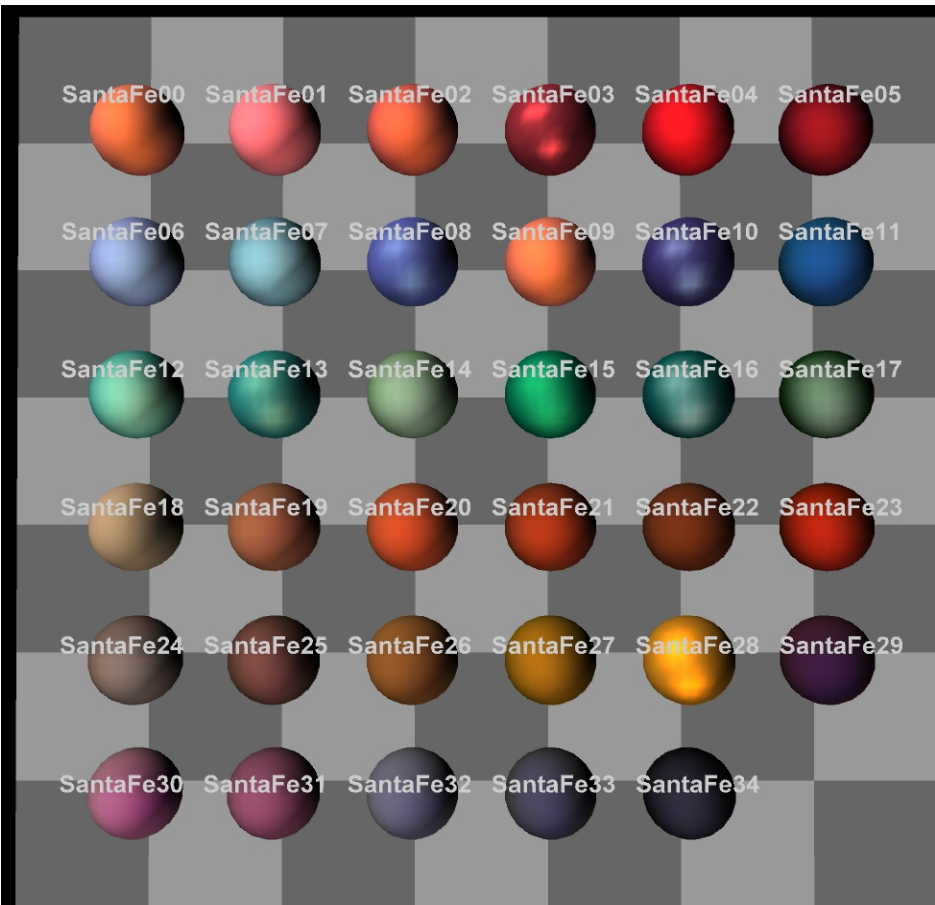
Selecting a Universal Media library and index number resets Material field values


```
.x3d \.x3dv \ECMAScript SAI\  
<Material DEF='SantaFe09'  
  diffuseColor='0.914894  
  emissiveColor='0.0 0.0 0.  
  specularColor='0.345745  
  transparency='0.0'  
  ambientIntensity='0.255814'  
  shininess='0.12'  
  containerField='material'  
>
```


Universal Media libraries include

ArtDeco, Autumn, Glass, Metal, Neon, Rococo, SantaFe, Sheen, Silky, Spring, Summer, Tropical, Winter

<http://www.web3d.org/x3d/content/examples/Basic/UniversalMediaMaterials>



 Material	<p>Material specifies surface material properties for associated geometry nodes Material attributes are used by the VRML lighting equations during rendering.</p> <p>Hint: insert Shape and Appearance nodes before adding material.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
diffuseColor	<p>[diffuseColor: accessType inputOutput, type SFCOLOR CDATA "0.8 0.8 0.8"]</p> <p>[RGB color] how much direct, angle-dependent light is reflected from all light sources.</p> <p>Hint: only emissiveColor affects IndexedLineSet, LineSet and PointSet.</p>
emissiveColor	<p>[emissiveColor: accessType inputOutput, type SFCOLOR CDATA "0 0 0"]</p> <p>[RGB color] how much glowing light is emitted from this object.</p> <p>Hint: emissiveColors glow even when all lights are off.</p> <p>Hint: reset diffuseColor from default (.8 .8 .8) to (0 0 0) to avoid washout.</p> <p>Hint: only emissiveColor affects IndexedLineSet, LineSet and PointSet.</p> <p>Warning: bright emissiveColor values can wash out some textures.</p>
specularColor	<p>[specularColor: accessType inputOutput, type SFCOLOR CDATA "0 0 0"]</p> <p>[RGB color] specular highlights are brightness reflections (example: shiny spots on an apple).</p> <p>Interchange profile hint: this field may be ignored.</p>
shininess	<p>[shininess: accessType inputOutput, type SFFloat CDATA "0.2"]</p> <p>[0..1] low values provide soft specular glows, high values provide sharper, smaller highlights.</p> <p>Interchange profile hint: this field may be ignored.</p>
ambientIntensity	<p>[ambientIntensity: accessType inputOutput, type SFFloat CDATA "0.2"]</p> <p>[0..1] how much ambient omnidirectional light is reflected from all light sources.</p> <p>Interchange profile hint: this field may be ignored.</p>
transparency	<p>[transparency: accessType inputOutput, type SFFloat CDATA "0"]</p> <p>[0..1] how "clear" an object is: 1.0 is completely transparent, 0.0 is completely opaque.</p> <p>Interchange profile hint: transparency < .5 opaque, transparency > .5 transparent.</p>
containerField	<p>[containerField: NMTOKEN "material"]</p> <p>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>[class CDATA #IMPLIED]</p> <p>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>

FillProperties node

FillProperties specifies additional characteristics that can be applied to the material shading of geometry nodes

- Adds to basic effects of peer Material and texture

FillProperties is a new X3D node not in VRML97

- If backwards compatibility needed and FillProperties effects are critical, consider an additional secondary technique to also backup this functionality

Note: hatch effects are not affected by lighting


FillProperties fields

- *filled* is a boolean (true or false) field to indicate whether the material properties are filled in. Setting *filled*='false' can be useful to highlight hatching effects.
- *hatched* is another SFBool single-field boolean that turns hatching effects on or off. Hatching can be a helpful user-interaction technique to indicate selection or objects of interest.
- *hatchColor* is the color applied to hatching effects over the material surface. Be sure to use a color that distinguishes hatching from diffuseColor.
- *hatchStyle* codes follow on the next slide

FillProperties hatchStyle codes

(parentheses indicate optional support)

Enumeration Code	Hatch Pattern
1	Horizontal equally spaced parallel lines
2	Vertical equally spaced parallel lines
3	Positive slope equally spaced parallel lines
4	Negative slope equally spaced parallel lines
5	Horizontal/vertical crosshatch
6	Positive slope/negative slope crosshatch
7	(cast iron or malleable iron and general use for all materials)
8	(steel)
9	(bronze, brass, copper, and compositions)
10	(white metal, zinc, lead, babbitt, and alloys)
11	(magnesium, aluminum, and aluminum alloys)
12	(rubber, plastic, and electrical insulation)
13	(cork, felt, fabric, leather, and fibre)
14	(thermal insulation)
15	(titanium and refractory material)
16	(marble, slate, porcelain, glass, etc.)
17	(earth)
18	(sand)
19	(repeating dot)

 FillProperties	FillProperties indicates whether appearance is filled or hatched. Hatches are applied on top of the already rendered appearance of the node, and are not affected by lighting.
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: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
filled	[filled: accessType inputOutput, type SBool (true false) "true"] Whether or not associated geometry is filled.
hatched	[hatched: accessType inputOutput, type SBool (true false) "true"] Whether or not associated geometry is hatched.
hatchStyle	[hatchStyle: accessType inputOutput, type SInt32 CDATA "1"] hatchStyle selects a hatch pattern from International Register of Graphical Items. 1=Horizontal equally spaced parallel lines. 2=Vertical equally spaced parallel lines. 3=Positive slope equally spaced parallel lines. 4=Negative slope equally spaced parallel lines. 5=Horizontal/vertical crosshatch. 6=Positive slope/negative slope crosshatch. 7=(cast iron or malleable iron and general use for all materials). 8=(steel). 9=(bronze, brass, copper, and compositions). 10=(white metal, zinc, lead, babbitt, and alloys). 11=(magnesium, aluminum, and aluminum alloys). 12=(rubber, plastic, and electrical insulation). 13=(cork, felt, fabric, leather, and fibre). 14=(thermal insulation). 15=(titanium and refractory material). 16=(marble, slate, porcelain, glass, etc.). 17=(earth). 18=(sand). 19=(repeating dot).
hatchColor	[hatchColor: accessType inputOutput, type SColor CDATA "1 1 1"] Color of the hatch pattern.
containerField	[containerField: NMTOKEN "fillProperties"] 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.

LineProperties node

LineProperties specifies additional characteristics that can be applied to the material shading of geometry nodes

- Adds to basic effects of peer Material and texture

LineProperties is a new X3D node not in VRML97

- If backwards compatibility needed and FillProperties effects are critical, consider an additional secondary technique to also backup this functionality


LineProperties fields

- *applied* is an SFBool field to turn the line property effects on or off, which can be set up as a helpful user-interaction technique
- *linewidthScaleFactor* (note irregular capitalization) provides a multiplicative factor to scale the nominal X3D-browser line width
- *linetype* (note irregular capitalization) selects a line pattern, with allowed values listed on following slide

LineProperties *linetype* values

(parentheses indicate optional support)

Enumeration Code	linetype Pattern
1	Solid
2	Dashed
3	Dotted
4	Dashed-dotted
5	Dash-dot-dot
6	(single)
7	(single dot)
8	(double arrow)
9	(chain line)
10	(center line)
11	(hidden line)
12	(phantom line)
13	(break line 1)
14	(break line 2)
15	User-specified dash pattern

 LineProperties	LineProperties specifies additional properties applicable to all line geometry.
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 <code>_all_</code> other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
applied	[applied: accessType inputOutput, type SFBool (true false) "true"] Whether or not LineProperties are applied to associated geometry.
linetype	[linetype: accessType inputOutput, type SFInt32 CDATA "0"] linetype selects a line pattern, with solid default if defined value isn't supported. Values with guaranteed support are 1 Solid, 2 Dashed, 3 Dotted, 4 Dashed-dotted, 5 Dash-dot-dot. Optionally supported values are 6 single, 7 single dot, 8 double arrow, 10 chain line, 11 center line, 12 hidden line, 13 phantom line, 14 break line 1, 15 break line 2, 16 User-specified dash pattern.
linewidthScaleFactor	[linewidthScaleFactor: accessType inputOutput, type SFFloat CDATA "0"] linewidthScaleFactor is a scale factor multiplied by browser-dependent nominal linewidth, mapped to nearest available line width. Values zero or less provide minimum available line width.
containerField	[containerField: NMTOKEN "lineProperties"] 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.

Texture nodes

Texture nodes read 2D image (or movie) files and apply them pixel-by-pixel to the associated geometry sharing the same Shape node

- Thus wrapping picture images around an object
- ImageTexture, PixelTexture, MovieTexture
- Can be inexpensive way to achieve high fidelity

Texture images can be shifted, rotated, scaled

- TextureTransform, TextureCoordinate
- Thus modifying image application to geometry

Texture coordinates 1

Defined by a 2D (s, t) coordinate system

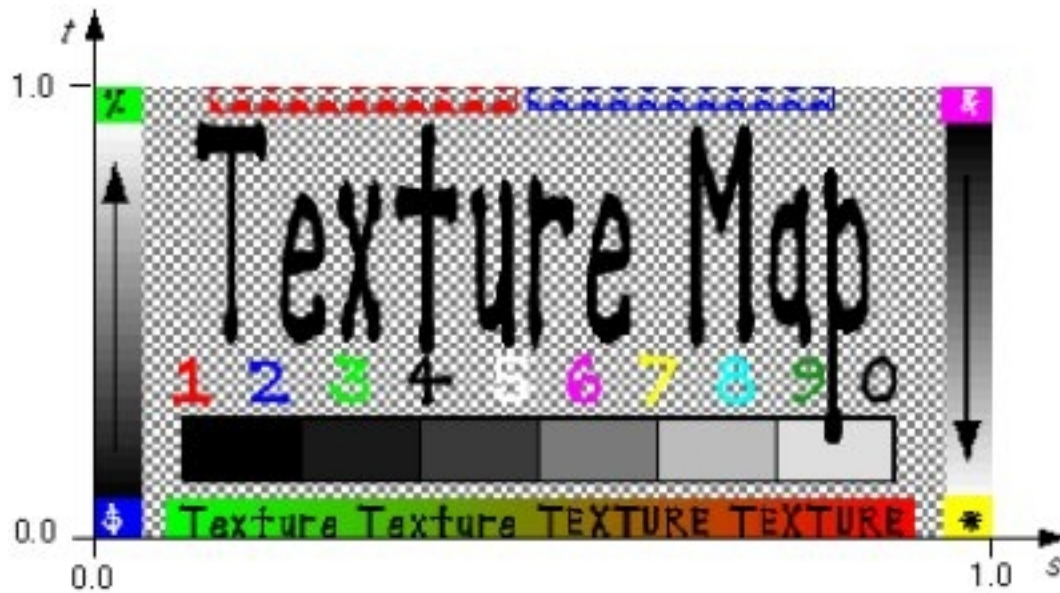
- Ranges from $[0,1]$ along lateral s and vertical t axes
- Bottom edge of image is s -axis ($t=0$)
- Left edge of image is t -axis ($s=0$)
- Top-right corner is $(s, t) = (1, 1)$

Thus texture maps provide a 2D color function that find the pixel in an image at location (s, t) to return value of $color(s, t)$

Texture coordinates 2

s and t coordinates locate each pixel in an image

- Thus texture coordinates work independently of either file size (bytes), image size (pixel count) or aspect ratio (width:height)



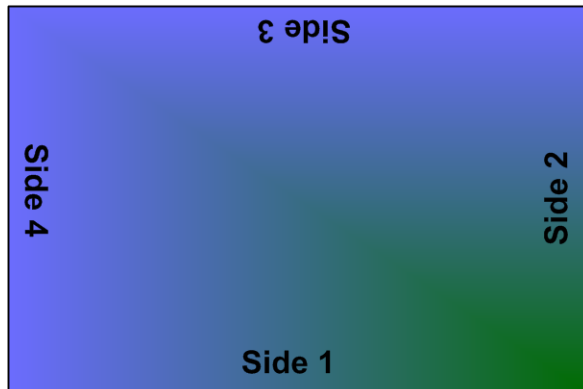
Common fields for texture nodes

repeatS and *repeatT*

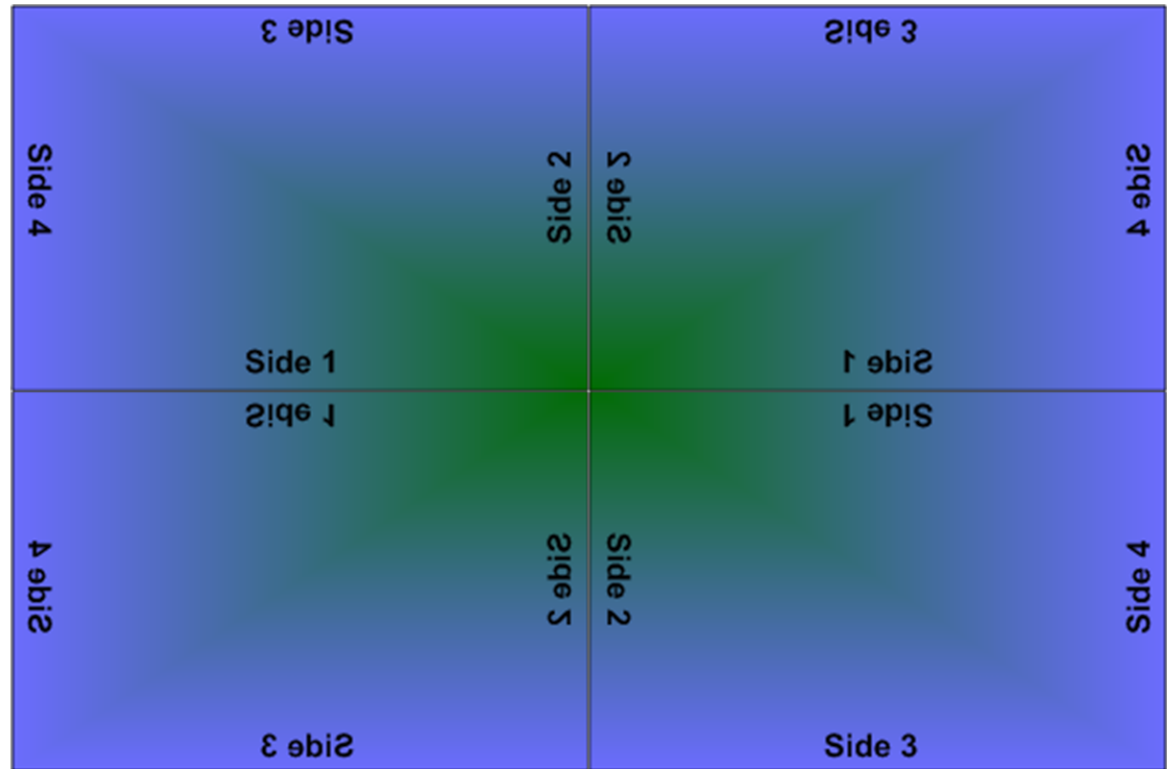
- These fields indicate whether the texture image is repeated along a given axis once used
- Default is to use once along each axis, mapping the texture image once from coordinates (0,0) to (1,1)

Texture flipping for (s,t) tile repetition

a. Original image



b. First flip copy of first image across rightmost edge



d. Note that all internal sides match as mirror images of each other.

Also note that external sides match:
top and bottom edges are both Side 3,
left and right edges are both Side 4.

Thus further (s,t) repetition also matches
when additional texture tiling occurs.

c. Then flip copy of both images across bottommost edge

Image file manipulation tools

Many tools are available for manipulating images, sometimes provided with the operating system

- Adobe Photoshop
- Microsoft Visio, Paint

One of best is free, open source, recommended:

- Gnu Image Manipulation Program (GIMP)
<http://www.gimp.org>

Drawing tools can also be helpful

- OpenOffice Draw, Impress
<http://www.openOffice.org>

ImageTexture node

ImageTexture retrieves a 2D image file and applies it as a texture to geometry

- Commonly used technique, important to master *url* described in Chapter 4 Grouping Nodes
- as part of Inline and Anchor
- Recall that the url field is an ordered list which can include both local (relative) and online addresses to image files
- Might preferentially load online version first, perhaps if it can be updated, and keep a local url value for a backup image

ImageTexture file formats

Supported, required image file formats:

- Joint Photographic Expert Group (.jpg) which is good for photographic images
- Portable Network Graphics (.png) which is good for bit-mapped drawings and other images
- Both formats are royalty free, commonly used in Web

Also recommended (but not required)

- Graphics Image Format (.gif), has license restrictions


Other image formats are also allowed

- but support by X3D browser not guaranteed

ImageTexture and Material

It is good practice to accompany ImageTexture with a Material node

- Material is rendered first if network delays are encountered when loading the image file
- Carefully chosen Material *diffuseColor* can reduce sudden color changes when a delayed image file is finally applied
- Underlying Material values are further important and will show through if the texture image includes transparent pixels

 ImageTexture	<p>ImageTexture maps a 2D-image file onto a geometric shape. Texture maps have a 2D coordinate system (s, t) horizontal and vertical, with (s, t) values in range [0.0, 1.0] for opposite corners of the image.</p> <p>Hint: insert Shape and Appearance nodes before adding texture.</p> <p>Warning: bright Material emissiveColor values can wash out some textures.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
url	<p>[url: accessType inputOutput, type MFString CDATA #IMPLIED]</p> <p>Location and filename of image. Multiple locations are more reliable, and Web locations let e-mail attachments work.</p> <p>Hint: Strings can have multiple values, so separate each string by quote marks ["http://www.url1.org" "http://www.url2.org" "etc."].</p> <p>Hint: XML encoding for " is &quot; (a character entity).</p> <p>Warning: strictly match directory and filename capitalization for http links!</p> <p>Hint: can replace embedded blank(s) in url queries with %20 for each blank character.</p>
repeatS	<p>[repeatS: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Horizontally repeat texture along S axis.</p>
repeatT	<p>[repeatT: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Vertically repeat texture along T axis.</p>
containerField	<p>[containerField: NMTOKEN "texture"]</p> <p>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>[class CDATA #IMPLIED]</p> <p>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>

MovieTexture node

MovieTexture applies video imagery to geometry

- Same considerations for *url* field as ImageTexture

Usually applied sparingly, because

- movie files are often quite large
- Applying high-frame rate pixels is computationally expensive, which can slow down frame rate and may present low-quality results in some browsers

3D wall or billboard helps make movies viewable

- Important to provide a clear Viewpoint to see it
- Authors might prefer for movie to instead play within Web browser outside X3D scene

MovieTexture fields 1

- *speed* is a rate factor to speed up or slow down movie playback, can be negative to go in reverse
- *startTime* and *stopTime* are used as input controls to begin and end play, usually by routing an SFTIME event from a TimeSensor or TouchSensor
- *pauseTime* and *resumeTime* operate similarly, allowing the movie to pause/resume at same point in time (rather than starting over from beginning)
- *isActive* and *isPaused* are boolean output events that are sent by the MovieTexture node: true when the condition occurs, false when it ends

MovieTexture fields 1

- *duration_changed* is length of time in seconds for one cycle of the movie
- *elapsedTime* is SFTIME output event sent continuously as movie is playing, cumulatively in seconds without counting any pause durations

Can use LoadSensor (chapter 12) to detect when movie is fully loaded

DEF and USE are important for multiple copies

- Minimize download file size, bandwidth, and delay

DEF MovieTexture	<p>MovieTexture applies a 2D movie image to surface geometry, or provides audio for a Sound node. First define as texture, then USE as Sound source to see it/hear it/save memory. Texture maps have a 2D coordinate system (s, t) horizontal and vertical, with (s, t) values in range [0.0, 1.0] for opposite corners of the image.</p> <p>Hint: insert Shape and Appearance nodes before adding texture.</p> <p>Hint: provide a viewpoint that allows a clear view of a MovieTexture so that users can easily see all details.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
url	<p>[url: accessType inputOutput, type MFString CDATA #IMPLIED]</p> <p>Location and filename of image Multiple locations are more reliable, and Web locations let e-mail attachments work.</p> <p>Hint: Strings can have multiple values, so separate each string by quote marks ["http://www.url1.org" "http://www.url2.org" "etc."].</p> <p>Hint: XML encoding for " is &quot; (a character entity).</p> <p>Warning: strictly match directory and filename capitalization for http links!</p> <p>Hint: can replace embedded blank(s) in url queries with %20 for each blank character.</p>
loop	<p>[loop: accessType inputOutput, type SFBool (true false) "false"]</p> <p>Repeat indefinitely when loop=true, repeat only once when loop=false.</p>
speed	<p>[speed: accessType inputOutput, type SFFloat CDATA "1.0"]</p> <p>Factor for how fast the movie (or soundtrack) is played.</p>
startTime	<p>[startTime: accessType inputOutput, type SFTIME CDATA "0"]</p> <p>Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT.</p> <p>Hint: usually receives a ROUTED time value.</p>
stopTime	<p>[stopTime: accessType inputOutput, type SFTIME CDATA "0"]</p> <p>Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT.</p> <p>Hint: usually receives a ROUTED time value.</p>
repeatS	<p>[repeatS: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Horizontally repeat texture along S axis.</p>
repeatT	<p>[repeatT: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Vertically repeat texture along T axis.</p>
duration_changed	<p>[duration_changed: accessType outputOnly, type SFTIME CDATA #FIXED ""]</p> <p>Length of time in seconds for one cycle of movie.</p>

isActive	<p>[isActive: outputOnly SFBoolLabel; #FIXED ""]</p> <p>isActive true/false events are sent when playback starts/stops.</p>
isPaused	<p>[isPaused: accessType outputOnly, type SFBool (true false) #FIXED ""]</p> <p>isPaused true/false events are sent when MovieTexture is paused/resumed.</p> <p>Warning: not supported in VRML97.</p>
pauseTime	<p>[pauseTime: accessType inputOutput, type SFTIME CDATA "0"]</p> <p>When time now >= pauseTime, isPaused becomes true and MovieTexture becomes paused. Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT.</p> <p>Hint: usually receives a ROUTED time value.</p> <p>Warning: not supported in VRML97.</p>
resumeTime	<p>[resumeTime: accessType inputOutput, type SFTIME CDATA "0"]</p> <p>When resumeTime becomes <= time now, isPaused becomes false and MovieTexture becomes active. Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT.</p> <p>Hint: usually receives a ROUTED time value.</p> <p>Warning: not supported in VRML97.</p>
elapsedTime	<p>[elapsedTime: accessType outputOnly, type SFTIME CDATA #FIXED ""]</p> <p>Current elapsed time since MovieTexture activated/running, cumulative in seconds, and not counting any paused time.</p> <p>Warning: not supported in VRML97.</p>
containerField	<p>[containerField: NMTOKEN "texture"]</p> <p>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>[class CDATA #IMPLIED]</p> <p>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>

PixelFormat node

PixelFormat contains the bit pattern of an image

- Written out as set of numeric data within the node

This allows single X3D scene to embed imagery

- Which helps when delivering a self-sufficient scene
- However may increase overall file size

Numeric image data is encoded pixel by pixel,
using a special data type: SFImage

SFImage data type

First three data values:






- Number of width pixels in image
- Number of height pixels in image
- Number of component values in each pixel (0-4)

Appropriate number of pixel values follow

Component values

- 0 for no image, `<ImageTexture image='0 0 0' />`
- 1 for black-white intensity
- 2 for black-white intensity, transparency
- 3 for red-green-blue colors
- 4 for red-green-blue colors, transparency

SFImage examples

Components	SFImage Value	Description	Image
0	0 0 0	Empty image	
1	1 2 1, 0xFF 0x00	Intensity (black & white) example: checkerboard pattern	
2	2 1 2, 0xCCFF 0x2277	Intensity & transparency example	
3	2 4 3, 0xFF0000 0xFF00 0 0 0 0 0xFFFF 0xFFFF00	Red-green-blue (RGB) example	
4	3 2 4, 1 0 0 255, 0 1 0 255, 0 0 1 255, 1 0 0 127, 0 1 0 127, 0 0 1 127	Red-green-blue-alpha (RGBA) example	

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3 <X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/spec
4 <head>
19 <Scene>
20 <ExternProtoDeclare name='WhereAmI' url='../Chapter14-Prototypes/WhereAmI.x3d#WhereAmI' "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapte
21 <ProtoInstance name='WhereAmI'>
22 <Background skyColor='1 1 1'>
23 <Viewpoint description='Book Viewpoint' orientation='0.736 0.615 -0.284 -0.32' position='-2.9 1.64 9.33'>
24 <Transform translation='2 2 0'>
25 <Shape>
26 <Appearance>
27 <PixelTexture DEF='PixelColors' image='2 4 3 0xff0000 0xffff00 0x007700 0xff0077 0x0000ff 0xff7700 0x00ff77 0x888888'
28 repeatS='false' repeatT='false'>
29 </Appearance>
30 <Box/>
31 </Shape>
32 </Transform>
33 <Transform translation='-2 2 0'>
34 <Shape>
35 <Appearance>
36 <PixelTexture USE='PixelColors'>
37 </Appearance>
38 <Cone/>
39 </Shape>
40 </Transform>
41 <Transform translation='2 -2 0'>
42 <Shape>
43 <Appearance>
44 <PixelTexture USE='PixelColors'>
45 </Appearance>
46 <Cylinder/>
47 </Shape>
48 </Transform>
49 <Transform translation='-2 -2 0'>
50 <Shape>
51 <Appearance>
52 <PixelTexture USE='PixelColors'>
53 </Appearance>
54 <Sphere/>
55 </Shape>
56 </Transform>
57 </Scene>
58 </X3D>

```

Edit PixelTexture

DEF ☒ PixelColors USE ☐ PixelColors

containerField ☐ texture

image

number width pixels

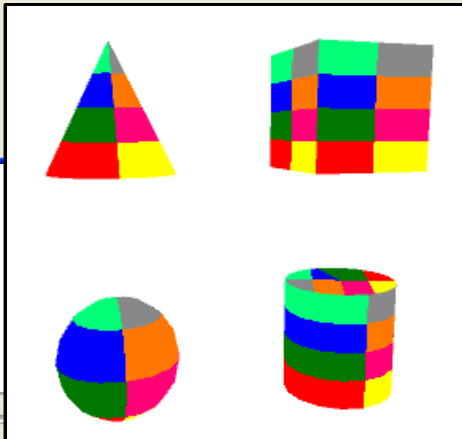
number height pixels

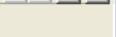
number color components

image data


repeatS ☐ repeatT ☐

OK Cancel Help





The screenshot shows a window titled "Xj3D Viewer". The main display area contains a 10x10 grid of squares in a checkerboard pattern, alternating between black and light gray. The top of the window has a blue title bar with a close button. The bottom of the window features a toolbar with various navigation icons (up, down, left, right, rotate, zoom, etc.) and a dropdown menu currently set to "Book Viewpoint". The status bar at the bottom right displays the number "58.82".

 PixelTexture	<p>PixelTexture creates a 2D-image texture map using a numeric array of pixel values. Texture maps have a 2D coordinate system (s, t) horizontal and vertical, with (s, t) values in range [0.0, 1.0] for opposite corners of the image.</p> <p>Hint: this is a good way to bundle image(s) into a single scene file, avoiding multiple downloads.</p> <p>Warning: aggregate file size can grow dramatically.</p> <p>Hint: insert Shape and Appearance nodes before adding texture.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
image	<p>[image: accessType inputOutput, type SFImage CDATA "0 0 0"]</p> <p>Defines image: width height number_of_components pixel_values. width and height are pixel count, number_of_components = 1 (intensity), 2 (intensity alpha), 3 (red green blue), 4 (red green blue alpha-transparency). intensity example: [1 2 1 0xFF 0x00] intensity-alpha example: [2 2 1 0 255 255 0] red-green-blue example: [2 4 3 0xFF0000 0xFF00 0 0 0 0 0xFF0000 0xFF0000] red-green-blue-alpha example: [needed]</p>
repeatS	<p>[repeatS: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Horizontally repeat texture along S axis.</p>
repeatT	<p>[repeatT: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Vertically repeat texture along T axis.</p>
containerField	<p>[containerField: NMTOKEN "texture"]</p> <p>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>[class CDATA #IMPLIED]</p> <p>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>

TextureTransform node

TextureTransform defines a 2D coordinate transformation for corresponding texture node, to better align images placed on geometry

- 2D translation left/right/up/down
- rotation angle about center
- 2D scaling, uniform or non-uniform

Transformation order remains significant

- translation, rotation, scale (same as Transform)
- However it is applied against coordinate system, **not** image file, so directions are counterintuitive

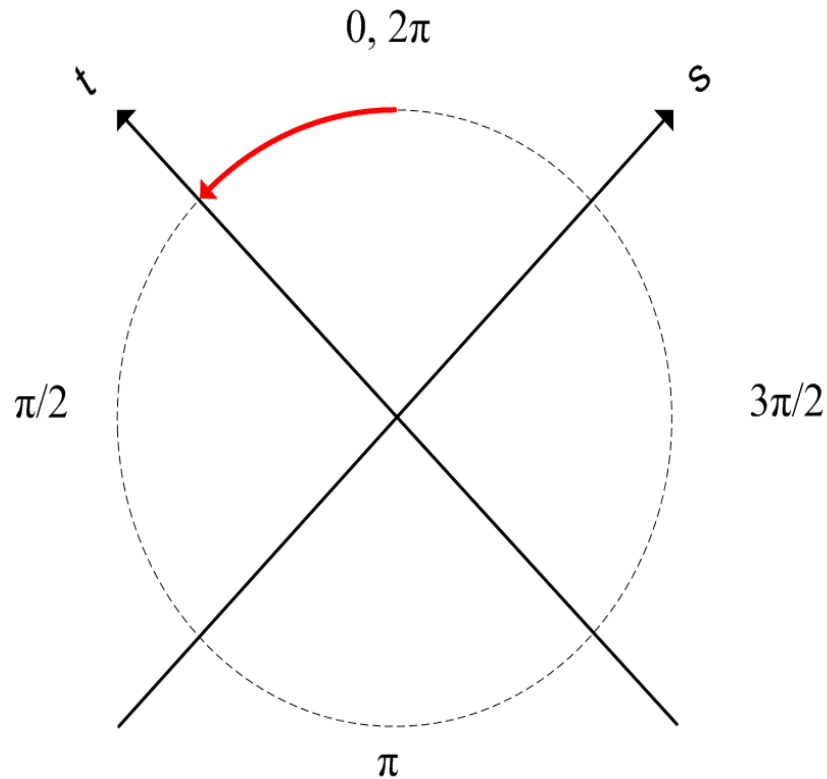
TextureTransform fields

Transformation are (s,t) axes-centric, not image centric, so direction differs from expectations

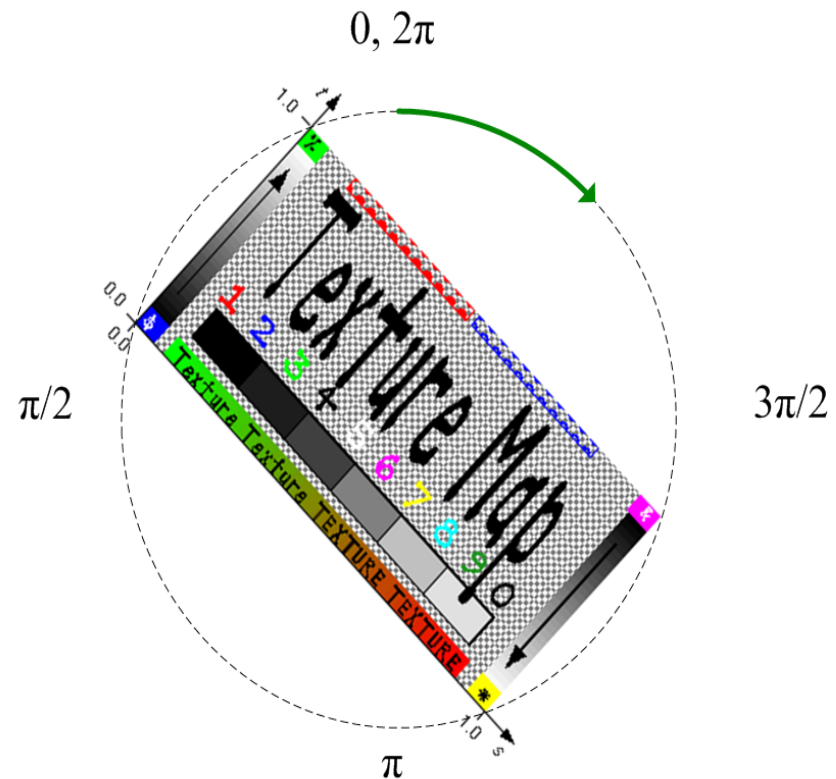
- *translation* controls lateral shift of image file along the polygonal surface, defined using (s,t) values
- *center* and *rotation* modify texture orientation: each makes a change in coordinate system, so the textured image rotates in opposite direction
 - *center* defined using (s,t) values
 - *rotation* defined using radians
- scale similarly opposite: *scale*='3 0.5' shows only 1/3 of texture along s axis, doubled along t axis

TextureTransform rotation

ccw rotation of
geometry texture coordinate axes



opposite rotation of
applied texture images

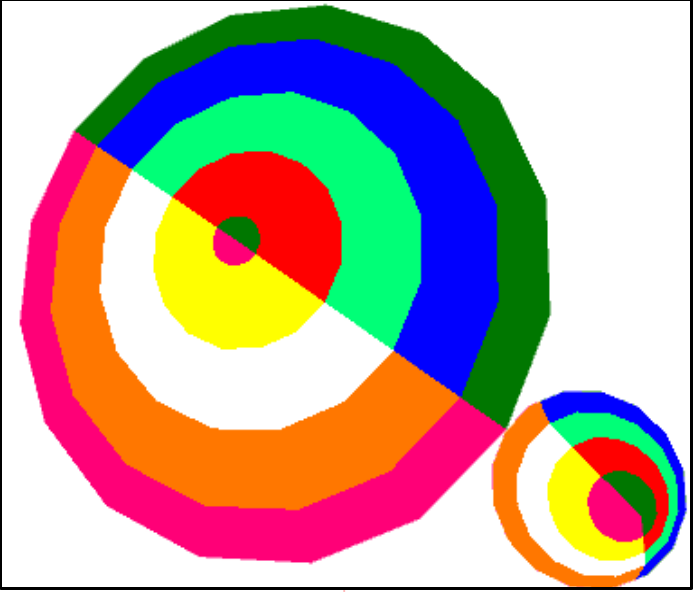


`<TextureTransform rotation='0.78' />`

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3 <X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/specifications/x3d-3.1.dtd'>
4   <head>
5     <meta content='TextureTransform.x3d' name='title' />
6     <meta content='An illustration of the same PixelTexture applied to a cone with different TextureTransform values' name='description' />
7     <meta content='Leonard Daly and Don Brutzman' name='creator' />
8     <meta content='2 February 2006' name='created' />
9     <meta content='2 February 2006' name='translated' />
10    <meta content='1 April 2007' name='modified' />
11    <meta content='http://X3dGraphics.com' name='reference' />
12    <meta content='http://www.web3d.org/x3d/content/examples/help.html' name='reference' />
13    <meta content='Copyright (c) 2006, Daly Realism and Don Brutzman' name='rights' />
14    <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dgraphics.com' name='subject' />
15    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter05-AppearanceMaterialTextures/TextureTransform.x3d' name='identifier' />
16    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
17    <meta content='../license.html' name='license' />
18  </head>
19  <Scene>
20    <ExternProtoDeclare name='WhereAmI' url='../Chapter14-Prototypes/WhereAmI.x3d#WhereAmI' "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter14-Prototypes/WhereAmI.x3d#WhereAmI">
21    <ProtoInstance name='WhereAmI' />
22    <Background skyColor='1 1 1' />
23    <Viewpoint description='Book View' orientation='-0.982 0.184 -0.044 1.37' position='0.99 6.24 1.57' />
24    <Transform translation='2 0 0'>
25      <Shape>
26        <Appearance>
27          <PixelTexture image='2 4 3'
28            0xff0000 0xffff00 0x007700 0xff0077
29            0x0000ff 0xff7700 0x00ff77 0xffffffff'
30            repeatS='true' repeatT='true' />
31          <TextureTransform translation='.33 .5' />
32        </Appearance>
33        <Cone height='3' />
34      </Shape>
35    </Transform>
36    <Transform translation='-2 0 0'>
37      <Shape>
38        <Appearance>
39          <PixelTexture image='2 4 3'
40            0xff0000 0xffff00 0x007700 0xff0077
41            0x0000ff 0xff7700 0x00ff77 0xffffffff'
42            repeatS='true' repeatT='true' />
43          <TextureTransform translation='.25 .33' />
44        </Appearance>
45        <Cone bottomRadius='3' height='1' />
46      </Shape>

```



Edit TextureTransform

☒ DEF
☐ USE

containerField


☐ textureTransform

center

rotation

translation

scale

 TextureTransform	<p>TextureTransform shifts 2D texture coordinates to position, orient and scale image patches. Visible effects appear reversed because image changes occur before mapping to geometry Order: translation, rotation about center, non-uniform scale about center.</p> <p>Hint: insert Shape and Appearance nodes before adding TextureTransform.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
translation	<p>[translation: accessType inputOutput, type SFVec2f CDATA "0 0"]</p> <p>Lateral/vertical shift in 2D (s,t) texture coordinates (opposite effect appears on geometry).</p>
center	<p>[center: accessType inputOutput, type SFVec2f CDATA "0 0"]</p> <p>center point in 2D (s,t) texture coordinates for rotation and scaling.</p>
rotation	<p>[rotation: accessType inputOutput, type SFFloat CDATA "0"]</p> <p>single rotation angle of texture about center (opposite effect appears on geometry).</p> <p>Warning: use a single radian angle value, not a 4-tuple Rotation.</p>
scale	<p>[scale: accessType inputOutput, type SFVec2f CDATA "1 1"]</p> <p>Non-uniform planar scaling of texture about center (opposite effect appears on geometry).</p>
containerField	<p>[containerField: NMTOKEN "textureTransform"]</p> <p>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>[class CDATA #IMPLIED]</p> <p>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>

TextureCoordinate node

TextureCoordinate specifies a set of 2D texture coordinates used by vertex-based nodes

- Such as IndexedFaceSet and ElevationGrid, which are covered in Chapter 6

TextureCoordinate *point* field has (s,t) values corresponding to vertices in parent geometry

- Type MFVec2f, multiple field array of 2-tuple floats
- Default is empty array, corresponds to regular (s,t) values ranging $(0,1)$

Best approach: use special authoring tools

TextureCoordinate	TextureCoordinate specifies 2D (s,t) texture-coordinate points, used by vertex-based geometry (ElevationGrid, IndexedFaceSet) to map textures to vertices (and patches to polygons). Hint: add Shape and then polygonal/planar geometry before adding TextureCoordinate.
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 <u>all</u> other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
point	[point: accessType inputOutput, type MFVec2f CDATA #IMPLIED] pairs of 2D (s,t) texture coordinates, either in range [0..1] or higher if repeating.
containerField	[containerField: NMTOKEN "texCoord"] 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.

TextureCoordinateGenerator node

TextureCoordinateGenerator enables the automatic computation and generation of texture coordinates for geometric shapes

- Can serve as substitute for TextureCoordinate node

Eleven procedural modes are provided

- *mode* field, following table explains possible values
- Associated *parameter* field provides setup values

This node is quite complicated and likely requires special authoring-tool support

- May also find support in 3D acceleration hardware

TextureCoordinateGenerator

mode enumerations and *parameter* values

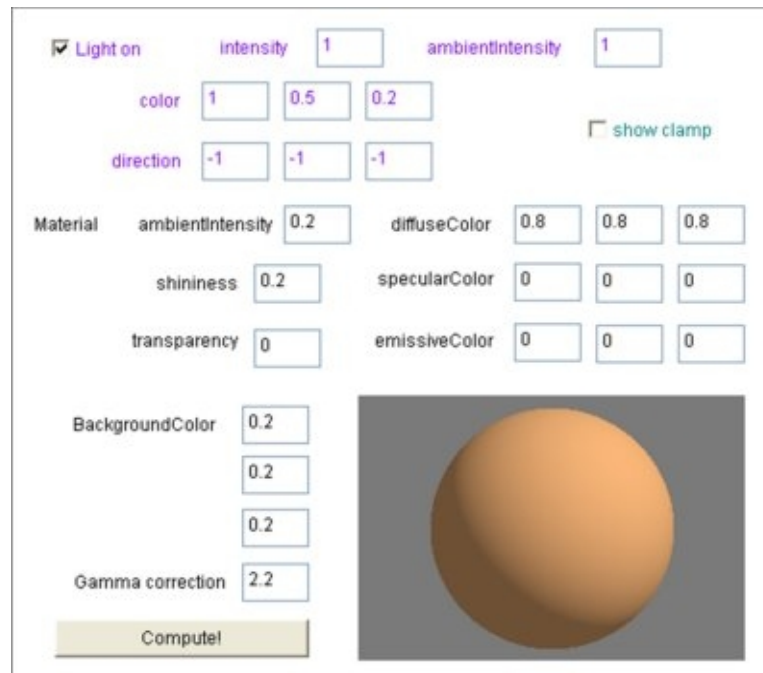
Mode	Description
SPHERE	Creates texture coordinates for a spherical environment or "chrome" mapping based on the vertex normals transformed to camera space. $u = N_x/2 + 0.5$ $v = N_y/2 + 0.5$ where u and v are the texture coordinates being computed, and N_x and N_y are the x and y components of the camera-space vertex normal. If the normal has a positive x component, the normal points to the right, and the u coordinate is adjusted to address the texture appropriately. Likewise for the v coordinate: positive y indicates that the normal points up. The opposite is of course true for negative values in each component. If the normal points directly at the camera, the resulting coordinates should receive no distortion. The $+0.5$ bias to both coordinates places the point of zero-distortion at the center of the sphere map, and a vertex normal of $(0, 0, z)$ addresses this point. Note that this formula doesn't take account for the z component of the normal.
CAMERASPACE NORMAL	Use the vertex normal, transformed to camera space, as input texture coordinates, resulting coordinates are in -1 to 1 range.
CAMERASPACE POSITION	Use the vertex position, transformed to camera space, as input texture coordinates
CAMERASPACE REFLECTIONVECTOR	Use the reflection vector, transformed to camera space, as input texture coordinates. The reflection vector is computed from the input vertex position and normal vector. $R = 2 \times \text{DotProd}(E, N) \times N - E$; In the preceding formula, R is the reflection vector being computed, E is the normalized position-to-eye vector, and N is the camera-space vertex normal. Resulting coordinates are in -1 to 1 range.
SPHERE - LOCAL	Sphere mapping but in local coordinates
COORD	Use vertex coordinates
COORD-EYE	Use vertex coordinates transformed to camera space
NOISE	Computed by applying Perlin solid noise function on vertex coordinates, parameter contains scale and translation [scale.x scale.y scale.z translation.x translation.y translation.z]
NOISE - EYE	Same as above but transform vertex coordinates to camera space first
SPHERE - REFLECT	Same as above but transform vertex coordinates to camera space first
SPHERE - REFLECT - LOCAL	Similar to "SPHERE - REFLECT", parameter[0] contains index of refraction, parameter[1 to 3] the eye point in local coordinates. By animating parameter [1 to 3] the reflection changes with respect to the point. Resulting coordinates are in -1 to 1 range.

Additional Resources

Additional Resources

Pellucid materials editor

- Eric Haines, copyright (c) 1997
- <http://tog.acm.org/resources/applets/vrml/pellucid.html>



Chapter Summary

Chapter Summary

Appearance affects associated geometry,
containing the following fields

Visual surface properties that interact with lights

- Material and TwoSidedMaterial
- LineProperties and FillProperties

Texture nodes wrap images onto geometry

- ImageTexture, MovieTexture, PixelTexture and MultiTexture
- TextureTransform, TextureCoordinate and TextureCoordinateGenerator

References

References 1

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

- Chapter 5, Appearance, Material and Textures
- <http://x3dGraphics.com>
- <http://x3dgraphics.com/examples/X3dForWebAuthors>

X3D Examples Help

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

References 2

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 10 – Materials
- Chapter 17 – Textures
- Chapter 18 – Texture Mapping
- Chapter 21 – Shiny Materials

Contact

Don Brutzman

brutzman@nps.edu

<http://web.nps.navy.mil/~brutzman>

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

1.831.656.2149 voice

1.831.656.7599 fax

Open-source license

Copyright (c) 1995-2008 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 5

Appearance, Material, and Textures

Things are not always as they appear.



Contents

Chapter Overview and Concepts

X3D Nodes and Examples

Additional Resources

Chapter Summary

References



Chapter Overview



Overview: Appearance, Material, and Textures

Appearance affects associated geometry,
containing the following fields

Visual surface properties that interact with lights

- Material and TwoSidedMaterial
- LineProperties and FillProperties

Texture nodes wrap images onto geometry

- ImageTexture, MovieTexture, PixelTexture and MultiTexture
- TextureTransform, TextureCoordinate and TextureCoordinateGenerator



[back to Table of Contents](#)

Concepts



Motivation

Appearance, material and texture nodes are intended to allow authors to make 3D objects look similar to objects in the real world

- This goal is always a worthy challenge

Lighting is an important factor in appearance, because 3D objects reflect their virtual light

- Appearance and lighting are computational
- In this chapter we assume white light available, usually from default NavigationInfo headlight
- Lighting and environment covered in Chapter 11



In a number of ways, lighting and appearance in 3D graphics are the theoretical inverses of optical properties. Graphics attempts to recreate optical properties computationally.

Parent-child constraints

Each Shape node can contain

- Single geometry node
- Single Appearance node

Each Appearance node can contain

- Single Material (or TwoSidedMaterial) node
- FillProperties, LineProperties, single Texture node

Each Texture node can contain

- Single TextureTransform or
TextureTransformGenerator node

Common functionality

Node repetition can be efficiently accomplished via DEF and USE

- Remember, first DEF must precede any USE copies
- Simplifies application of consistent coloring to multiple pieces of geometry which are either similar or parts of the same larger object

Consistent, more efficient, easier to globally change all instances at once

- Which is further important when changing styles or applying accessibility techniques throughout



[back to Table of Contents](#)

X3D Nodes and Examples



Appearance node

Each Shape contains some geometry along with a corresponding Appearance node

Appearance is a container which may include

- Single Material (or TwoSidedMaterial) node
- FillProperties, LineProperties, single Texture node

This close association makes assignment of rendering properties to geometry unambiguous

- Repetition of values for visual consistency is easily accomplished with DEF/USE of Appearance, Material, Texture node, etc.
- Clear naming helps, for example

```
<Appearance USE='FoggyGlassAppearance' />
```

DEF/USE names can get confusing in a large X3D scene, unless good patterns and habits are used when giving names to nodes.

For example, a DEF name of `FoggyGlass` certainly describes what is intended, but it is not clear whether the node is an Appearance, Material, or even some kind of Texture. Therefore, including the name of the defining node in the DEF name (e.g. `FoggyGlassAppearance`) makes it easy to copy.

In other words, it is more likely to later say

```
<Appearance USE='FoggyGlassAppearance' />
```

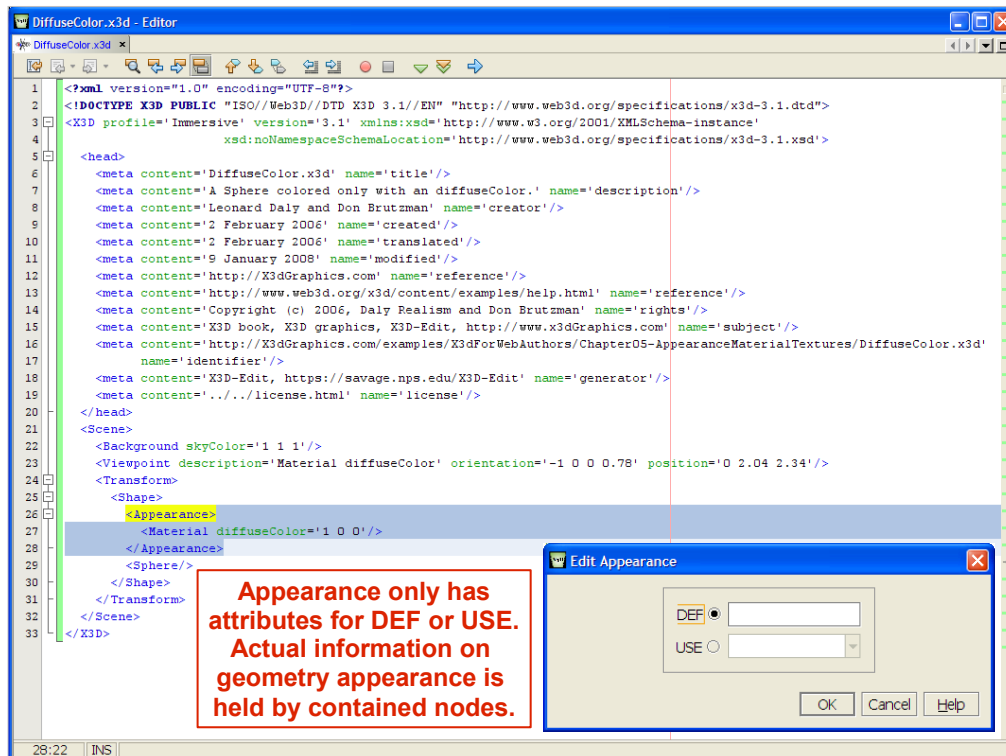
instead of making the node-typing mistake

```
<Material USE='FoggyGlass' /> <!-- run-time error -->
```

Since such run-time errors are often not caught until an end user is trying to view a scene with unintended errors, it is better to adopt good naming practices early to avoid puzzling problems later.

Thumbrules on node-naming conventions are given in the X3D Scene Authoring Hints, provided in the X3D-Edit help system and also online at

<http://www.web3d.org/x3d/content/examples/X3dSceneAuthoringHints.html#NamingConventions>



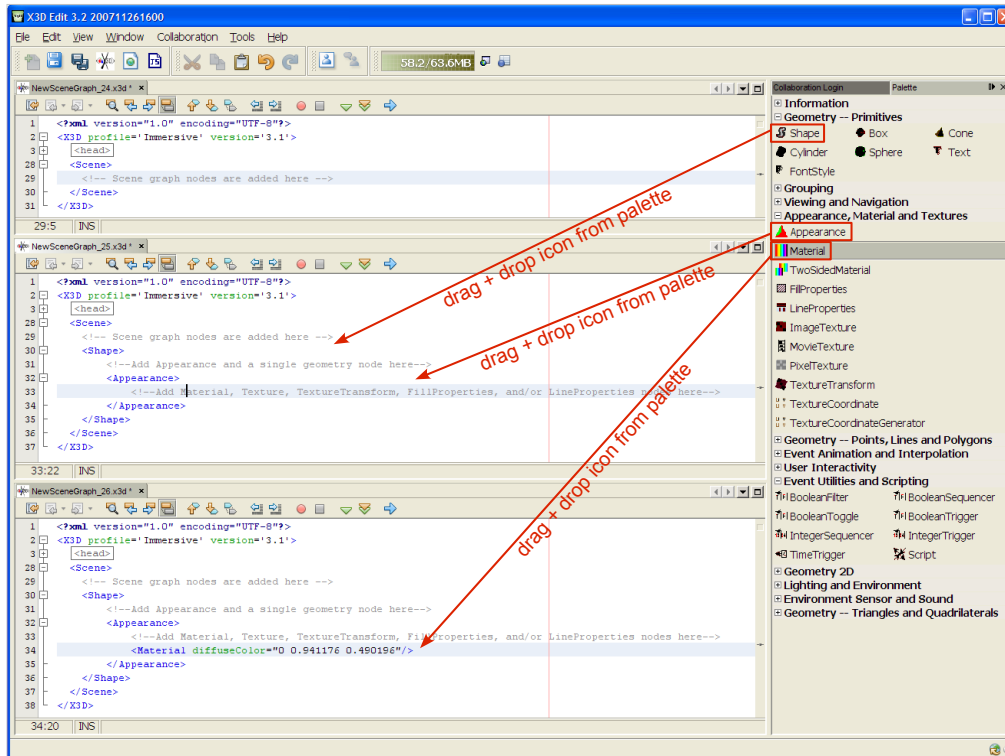
When adding an Appearance node from the palette, the following text appears in the editing pane, prompting further node addition(s):

```
<Appearance>
  <!--Add Material, Texture, TextureTransform, FillProperties, and/or
    LineProperties nodes here-->
</Appearance>
```

Palette simplifies addition of new nodes

Use X3D-Edit palette to pick the node of interest:

- Palette groups match chapter structure, and can be reordered by dragging with mouse
- Upon dragging a new node element into scene, corresponding node editor pops up
- After checking attribute values with node editor, select OK to confirm the new node
- Default attribute values omitted in XML for clarity
- Erroneous node placement in scene graph, or invalid attribute values, cause a validation error
- Accept or reject validation errors as appropriate, then continue with text editing if desired




Hint: place the cursor before comments and closing tags, and then press Enter (return key for line feeds), to get proper line spacing and to make the scene easier to read.

Embedded comments (that prompt where new nodes are inserted) can be deleted.

When all nodes are in place, you can reformat by selecting

- Control+A to select all nodes
- Alt+Shift+F to format the XML (also available via right-click context menu)

Note that head element is iconized and DOCTYPE deleted in these scenes for clarity.

 Appearance	Appearance specifies the visual properties of geometry by containing the Material, Texture and TextureTransform nodes. Hint: insert a Shape node before adding geometry or Appearance. Interchange profile hint: only Material and ImageTexture are allowed.
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: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
containerField	[containerField: NMTOKEN "appearance"] 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#Appearance>

Material node

Material controls how most geometry is colored, whether it is transparent or glowing, etc.

Surface visual properties are applied equally across all polygons making up a shape

Material properties define how geometry visually interacts with light sources in the scene

- Lighting and Environment is covered in Chapter 11
- Rendering results also depend on view perspective

Material is an important node to master



TwoSidedMaterial node

TwoSidedMaterial fields are identical to Material, with the addition of the following new fields:

- *backAmbientIntensity*
- *backDiffuseColor*, *backEmissiveColor*, *backSpecularColor*
- *backShininess*
- *backTransparency*

The 'back' fields determine how the 'backsides' of polygons are drawn

- Such as insides of primitive geometry
- Corresponding geometry must have *solid*='false'



TwoSidedMaterial was introduced in X3D version 3.2.

http://www.web3d.org/x3d/specifications/ISO-IEC-19775-X3DAbstractSpecification_Revision1_to_Part1/Part01/components/shape.html#TwoSidedMaterial

Reading X3D Specification node signatures

Actual X3D Specification entries are as follows:

- SFFloat [in,out] *backAmbientIntensity* 0.2 [0,1]
- SFCOLOR [in,out] *backDiffuseColor* 0.8 0.8 0.8 [0,1]
- SFCOLOR [in,out] *backEmissiveColor* 0 0 0 [0,1]
- SFFloat [in,out] *backShininess* 0.2 [0,1]
- SFCOLOR [in,out] *backSpecularColor* 0 0 0 [0,1]
- SFFloat [in,out] *backTransparency* 0 [0,1]

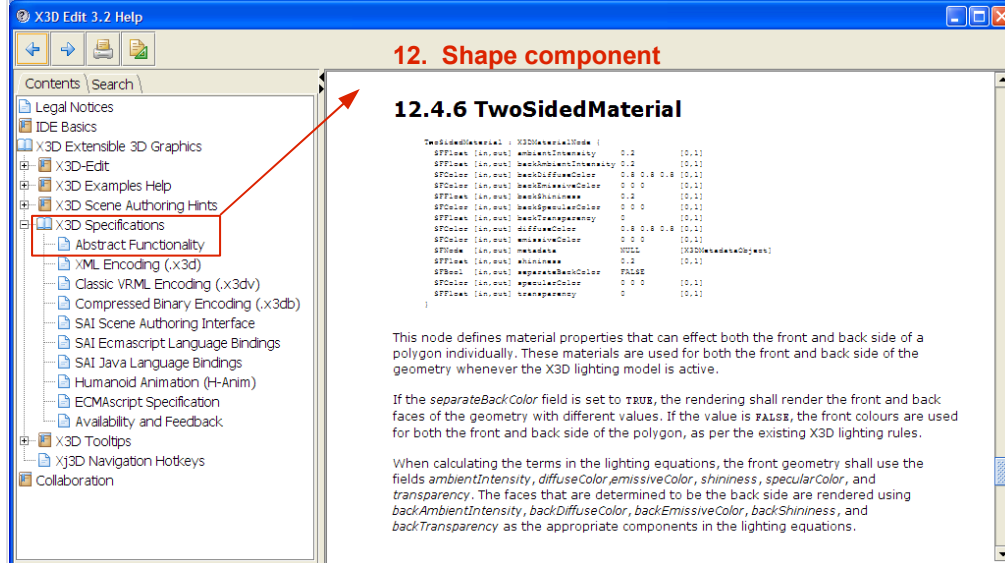
These field signatures are interpreted as follows:

- SFCOLOR and SFFloat are field types
- [in,out] is accessType
- default value is followed by [min,max]



This is a good time to look at the X3D specification entry for TwoSidedMaterial, either within X3D Help or online at

TwoSidedMaterial specification help entry in X3D-Edit

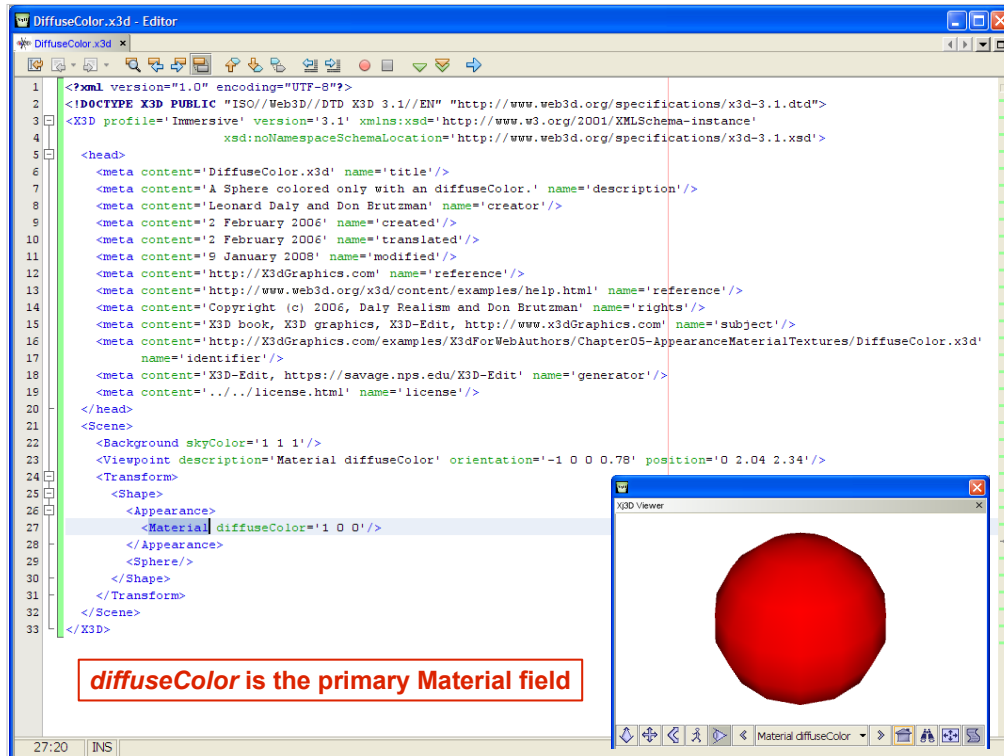


F1 is the X3D-Edit hot key to invoke the JavaHelp system

Material fields

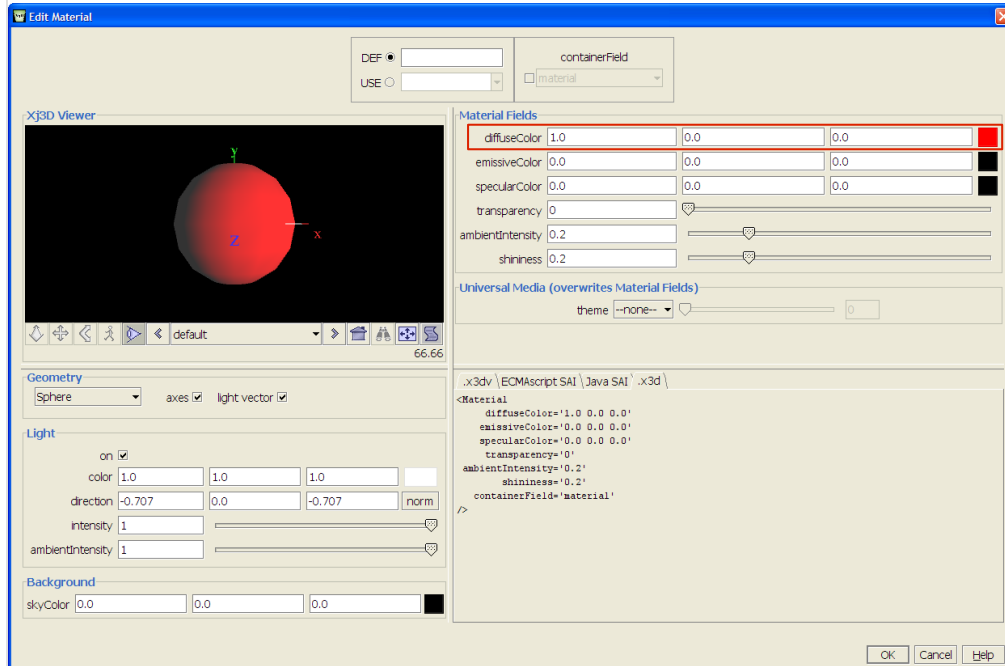
Color, transparency and shininess fields together make up Material properties. Examples follow.

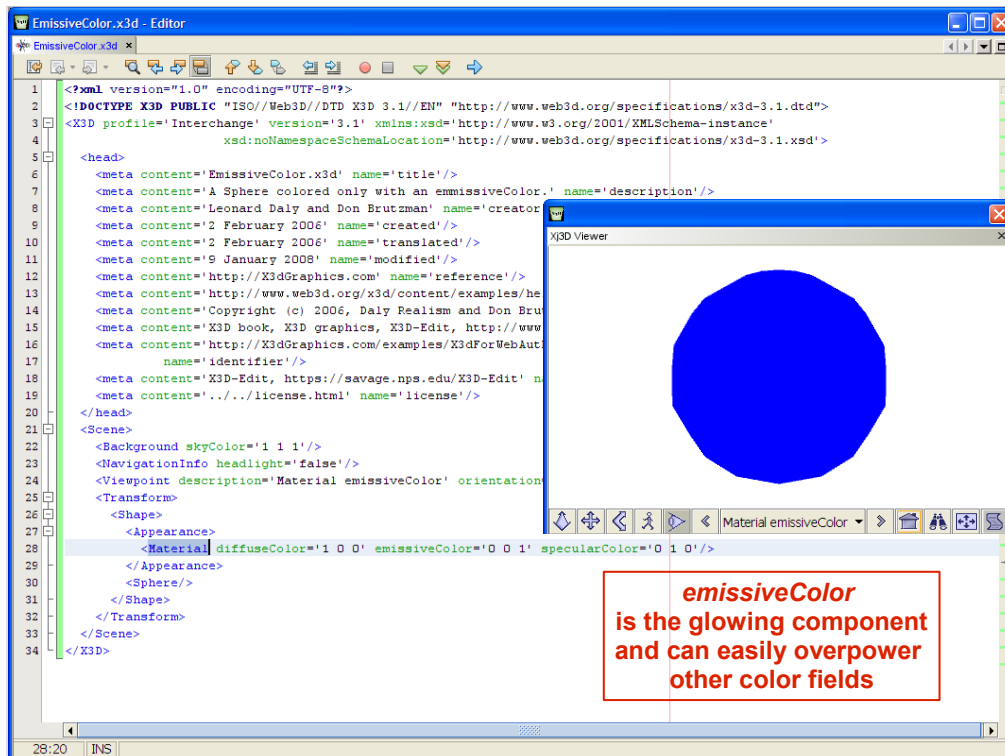
- *diffuseColor* reflects all X3D light sources, depending on viewing angles towards each light
- *ambientIntensity* is reflection multiplication factor
- *emissiveColor* is glowing component, normally off, independent of reflected light
- *specularColor* governs reflection highlights
- *shininess* controls specular intensity
- *transparency* is ability to see through an object:
1 is completely transparent, 0 is opaque



<http://x3dgraphics.com/examples/X3dForWebAuthors/Chapter05-AppearanceMaterialTextures/DiffuseColor.x3d>

Material editor: *diffuseColor*



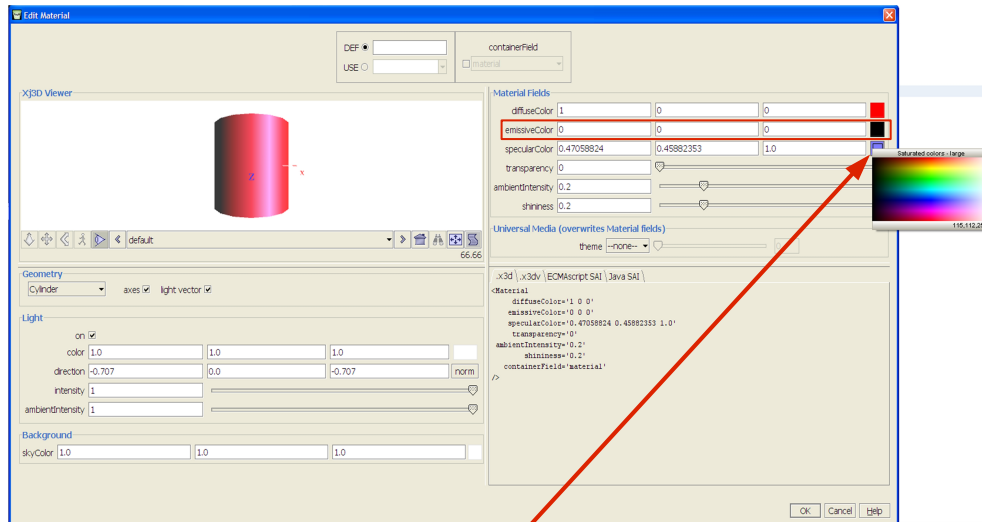


<http://x3dgraphics.com/examples/X3dForWebAuthors/Chapter05-AppearanceMaterialTextures/EmissiveColor.x3d>

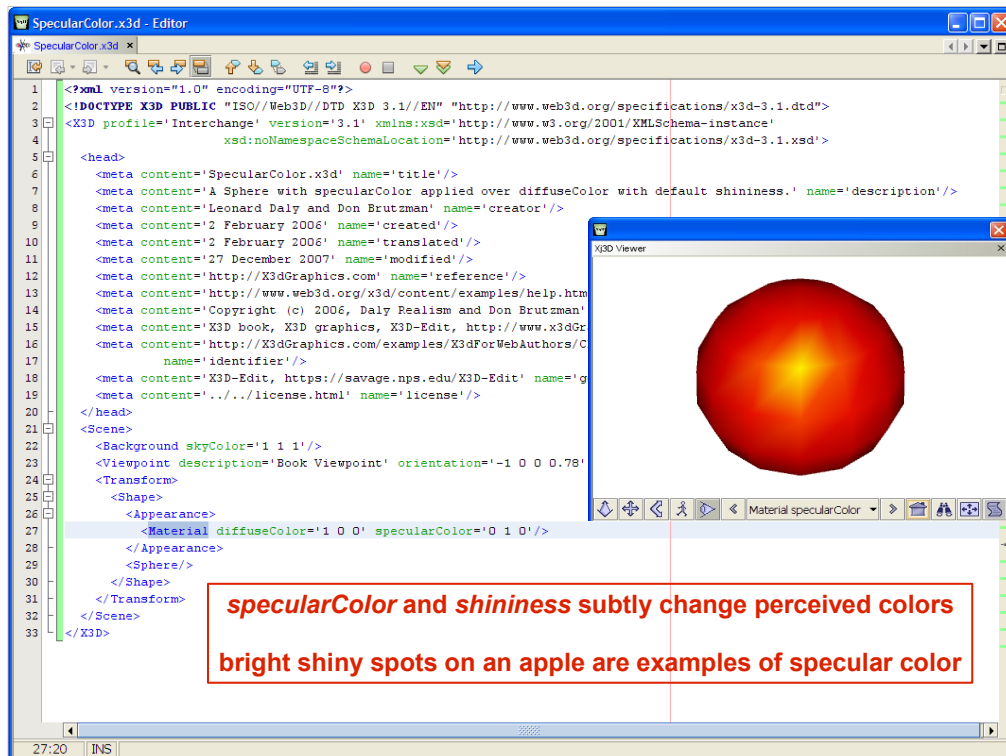
Also note how all highlights are washed out, the sense of perspective provided by the shading of reflected light is completely lost.

Because of this side effect, emissiveColor should be used sparingly (if at all) and is usually reserved for visualizing energy or other special effects.

Material editor color selector

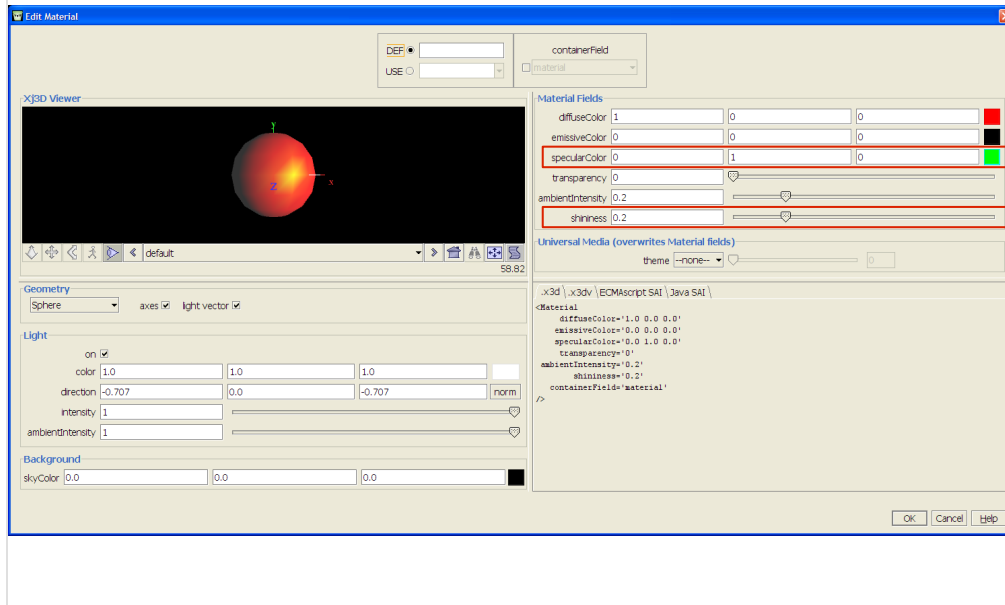


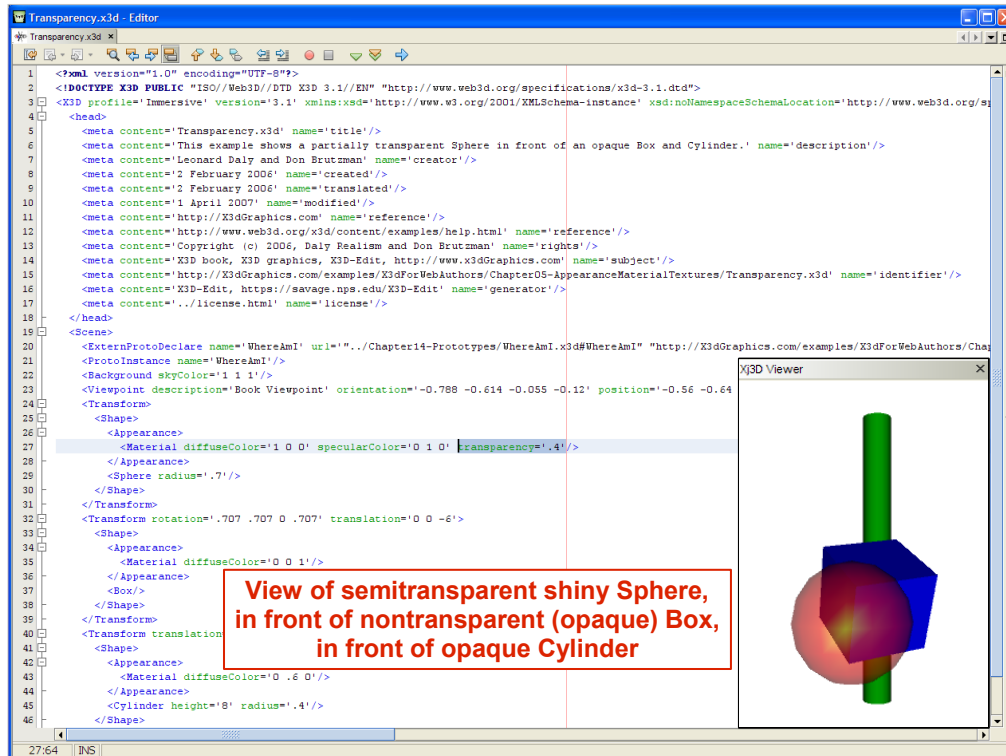
Click colored box to select a color



<http://x3dgraphics.com/examples/X3dForWebAuthors/Chapter05-AppearanceMaterialTextures/SpecularColor.x3d>

Material editor *specularColor, shininess*





<http://x3dgraphics.com/examples/X3dForWebAuthors/Chapter05-AppearanceMaterialTextures/Transparency.x3d>

Universal Media materials library

The Universal Media materials were originally created by SGI as part of OpenInventor in the 1990s as a convenience to authors

Each set of materials is grouped for visual compatibility and aesthetic appeal

Now converted and available for X3D use

- David Rousseau converted to VRML97
- Aaron Walsh created VRML Universal Media archive
- Don Brutzman translated into X3D as prototypes, cut/paste field values, also embedded in X3D-Edit
- <http://www.web3d.org/x3d/content/examples/Basic/UniversalMediaMaterials>

David Rousseau's VRML site for these materials is <http://vrm1stuff.free.fr/materials>

VRML Materials

VRML Materials

This site contains VRML 2.0 materials converted from the SGI's Open Inventor material examples.

Conversion example :

Inventor	VRML 2.0
<pre>#Inventor V1.0 ascii #artdeco00.iv Material { ambientColor 0.0706087 0.0212897 0.0336154 diffuseColor 0.282435 0.0851587 0.134462 specularColor 0.276305 0.11431 0.139857 emissiveColor 0 0 0 shininess 0.127273 transparency 0 }</pre>	<pre>#VRML V2.0 utf8 PROTO Artdeco00 [] { Material { ambientIntensity 0.250000 diffuseColor 0.282435 0.085159 0.134462 specularColor 0.276305 0.114310 0.139857 emissiveColor 0.000000 0.000000 0.000000 shininess 0.127273 transparency 0.000000 } }</pre>

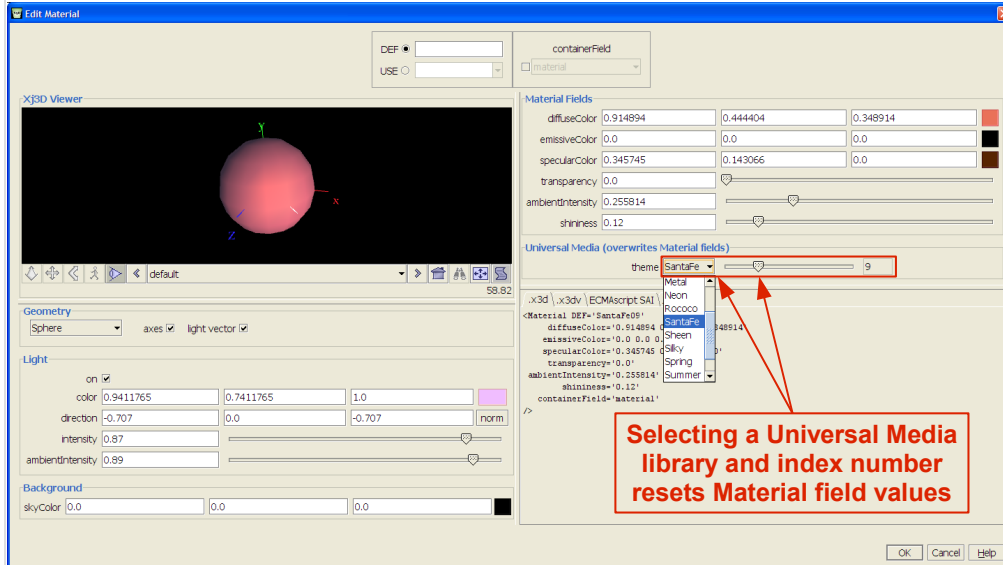
VRML2.0 "ambientIntensity" is calculated by the mean factor between the Inventor "ambientColor" and "diffuseColor".

How to use these materials :

VRML 2.0 supports **Prototypes** definitions as well as **External Prototypes**, the easiest way to use these materials is to gather the materials inside a separate file and declare an EXTERNPROTO in the file you want to use these materials (see example below).

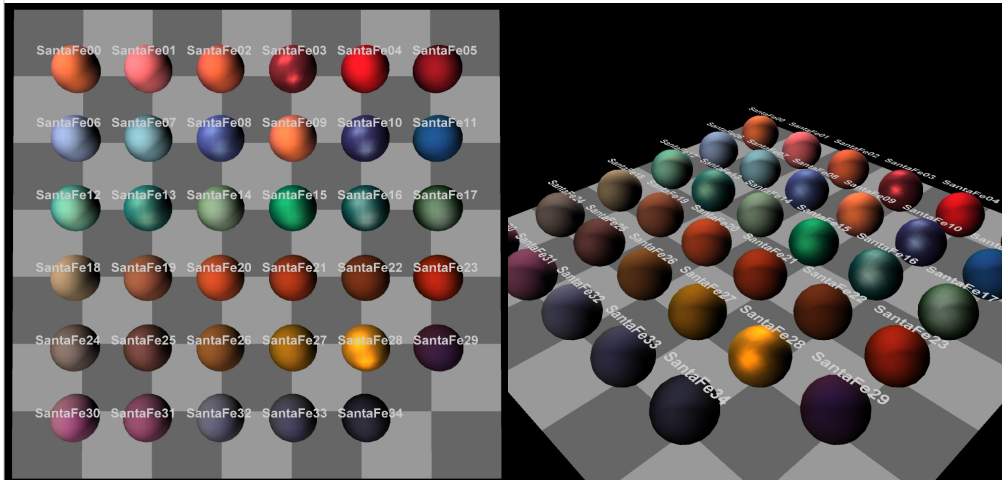
Unfortunately, most of the VRML 2.0 browsers doesn't seem to support external prototypes (especially PC versions), so you might as well cut and paste any material prototype inside the file you want to use it.


Selecting a Universal Material value



Universal Media libraries include
ArtDeco, Autumn, Glass, Metal, Neon, Rococo, SantaFe,
Sheen, Silky, Spring, Summer, Tropical, Winter

<http://www.web3d.org/x3d/content/examples/Basic/UniversalMediaMaterials>



 Material	Material specifies surface material properties for associated geometry nodes. Material attributes are used by the VRML lighting equations during rendering. Hint: insert Shape and Appearance nodes before adding material.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referenceable 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: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
diffuseColor	[diffuseColor: accessType inputOutput, type SFCOLOR CDATA "0.8 0.8 0.8"] [RGB color] how much direct, angle-dependent light is reflected from all light sources. Hint: only emissiveColor affects IndexedLineSet, LineSet and PointSet.
emissiveColor	[emissiveColor: accessType inputOutput, type SFCOLOR CDATA "0 0 0"] [RGB color] how much glowing light is emitted from this object. Hint: emissiveColors glow even when all lights are off. Hint: reset diffuseColor from default (.8 .8 .8) to (0 0 0) to avoid washout. Hint: only emissiveColor affects IndexedLineSet, LineSet and PointSet. Warning: bright emissiveColor values can wash out some textures.
specularColor	[specularColor: accessType inputOutput, type SFCOLOR CDATA "0 0 0"] [RGB color] specular highlights are brightness reflections (example: shiny spots on an apple). Interchange profile hint: this field may be ignored.
shininess	[shininess: accessType inputOutput, type SFFloat CDATA "0.2"] [0..1] low values provide soft specular glows, high values provide sharper, smaller highlights. Interchange profile hint: this field may be ignored.
ambientIntensity	[ambientIntensity: accessType inputOutput, type SFFloat CDATA "0.2"] [0..1] how much ambient omnidirectional light is reflected from all light sources. Interchange profile hint: this field may be ignored.
transparency	[transparency: accessType inputOutput, type SFFloat CDATA "0"] [0..1] how "clear" an object is: 1.0 is completely transparent, 0.0 is completely opaque. Interchange profile hint: transparency < .5 opaque, transparency > .5 transparent.
containerField	[containerField: NMTOKEN "material"] 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#Material>

FillProperties node

FillProperties specifies additional characteristics that can be applied to the material shading of geometry nodes

- Adds to basic effects of peer Material and texture

FillProperties is a new X3D node not in VRML97

- If backwards compatibility needed and FillProperties effects are critical, consider an additional secondary technique to also backup this functionality

Note: hatch effects are not affected by lighting



FillProperties fields

- *filled* is a boolean (true or false) field to indicate whether the material properties are filled in. Setting *filled*='false' can be useful to highlight hatching effects.
- *hatched* is another SFBool single-field boolean that turns hatching effects on or off. Hatching can be a helpful user-interaction technique to indicate selection or objects of interest.
- *hatchColor* is the color applied to hatching effects over the material surface. Be sure to use a color that distinguishes hatching from *diffuseColor*.
- *hatchStyle* codes follow on the next slide




FillProperties hatchStyle codes (parentheses indicate optional support)

Enumeration Code	Hatch Pattern
1	Horizontal equally spaced parallel lines
2	Vertical equally spaced parallel lines
3	Positive slope equally spaced parallel lines
4	Negative slope equally spaced parallel lines
5	Horizontal/vertical crosshatch
6	Positive slope/negative slope crosshatch
7	(cast iron or malleable iron and general use for all materials)
8	(steel)
9	(bronze, brass, copper, and compositions)
10	(white metal, zinc, lead, babbitt, and alloys)
11	(magnesium, aluminum, and aluminum alloys)
12	(rubber, plastic, and electrical insulation)
13	(cork, felt, fabric, leather, and fibre)
14	(thermal insulation)
15	(titanium and refractory material)
16	(marble, slate, porcelain, glass, etc.)
17	(earth)
18	(sand)
19	(repeating dot)

web|3D
CONSORTIUM



X3D for Web Authors, Table 5.8, p.136

 FillProperties	FillProperties indicates whether appearance is filled or hatched. Hatches are applied on top of the already rendered appearance of the node, and are not affected by lighting.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. <i>Hint:</i> descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children. <i>Hint:</i> USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
filled	[filled: accessType inputOutput, type SFBool (true false) "true"] Whether or not associated geometry is filled.
hatched	[hatched: accessType inputOutput, type SFBool (true false) "true"] Whether or not associated geometry is hatched.
hatchStyle	[hatchStyle: accessType inputOutput, type SInt32 CDATA "1"] hatchStyle selects a hatch pattern from International Register of Graphical Items. 1=Horizontal equally spaced parallel lines. 2=Vertical equally spaced parallel lines. 3=Positive slope equally spaced parallel lines. 4=Negative slope equally spaced parallel lines. 5=Horizontal/vertical crosshatch. 6=Positive slope/negative slope crosshatch. 7=(cast iron or malleable iron and general use for all materials). 8=(steel). 9=(bronze, brass, copper, and compositions). 10=(white metal, zinc, lead, babbitt, and alloys). 11=(magnesium, aluminum, and aluminum alloys). 12=(rubber, plastic, and electrical insulation). 13=(cork, felt, fabric, leather, and fibre). 14=(thermal insulation). 15=(titanium and refractory material). 16=(marble, slate, porcelain, glass, etc.). 17=(earth). 18=(sand). 19=(repeating dot).
hatchColor	[hatchColor: accessType inputOutput, type SFColor CDATA "1 1 1"] Color of the hatch pattern.
containerField	[containerField: NMTOKEN "fillProperties"] 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#FillProperties>

LineProperties node

LineProperties specifies additional characteristics that can be applied to the material shading of geometry nodes

- Adds to basic effects of peer Material and texture

LineProperties is a new X3D node not in VRML97

- If backwards compatibility needed and FillProperties effects are critical, consider an additional secondary technique to also backup this functionality



LineProperties fields


- *applied* is an SFBool field to turn the line property effects on or off, which can be set up as a helpful user-interaction technique
- *linewidthScaleFactor* (note irregular capitalization) provides a multiplicative factor to scale the nominal X3D-browser line width
- *linetype* (note irregular capitalization) selects a line pattern, with allowed values listed on following slide

LineProperties *linetype* values (parentheses indicate optional support)

Enumeration Code	linetype Pattern
1	Solid
2	Dashed
3	Dotted
4	Dashed-dotted
5	Dash-dot-dot
6	(single)
7	(single dot)
8	(double arrow)
9	(chain line)
10	(center line)
11	(hidden line)
12	(phantom line)
13	(break line 1)
14	(break line 2)
15	User-specified dash pattern



X3D for Web Authors, Table 5.11, p.138

 LineProperties	LineProperties specifies additional properties applicable to all line geometry.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. <i>Hint:</i> 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. <i>Hint:</i> USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
applied	[applied: accessType inputOutput, type SBool (true/false) "true"] Whether or not LineProperties are applied to associated geometry.
linetype	[linetype: accessType inputOutput, type SInt32 CDATA "0"] linetype selects a line pattern, with solid default if defined value isn't supported. Values with guaranteed support are 1 Solid, 2 Dashed, 3 Dotted, 4 Dashed-dotted, 5 Dash-dot-dot. Optionally supported values are 6 single, 7 single dot, 8 double arrow, 10 chain line, 11 center line, 12 hidden line, 13 phantom line, 14 break line 1, 15 break line 2, 16 User-specified dash pattern.
linewidthScaleFactor	[linewidthScaleFactor: accessType inputOutput, type SFloat CDATA "0"] linewidthScaleFactor is a scale factor multiplied by browser-dependent nominal linewidth, mapped to nearest available line width. Values zero or less provide minimum available line width.
containerField	[containerField: NMTOKEN "lineProperties"] 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#LineProperties>

Texture nodes

Texture nodes read 2D image (or movie) files and apply them pixel-by-pixel to the associated geometry sharing the same Shape node

- Thus wrapping picture images around an object
- [ImageTexture](#), [PixelTexture](#), [MovieTexture](#)
- Can be inexpensive way to achieve high fidelity

Texture images can be shifted, rotated, scaled

- [TextureTransform](#), [TextureCoordinate](#)
- Thus modifying image application to geometry

Texture coordinates 1

Defined by a 2D (s, t) coordinate system

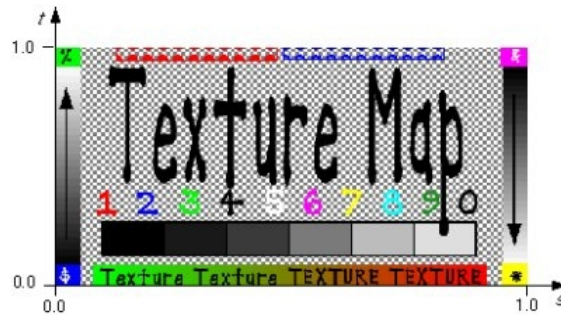
- Ranges from $[0,1]$ along lateral s and vertical t axes
- Bottom edge of image is s -axis ($t=0$)
- Left edge of image is t -axis ($s=0$)
- Top-right corner is $(s, t) = (1, 1)$

Thus texture maps provide a 2D color function that find the pixel in an image at location (s, t) to return value of $color(s, t)$

Texture coordinates 2

s and t coordinates locate each pixel in an image

- Thus texture coordinates work independently of either file size (bytes), image size (pixel count) or aspect ratio (width:height)



web|3D
CONSORTIUM



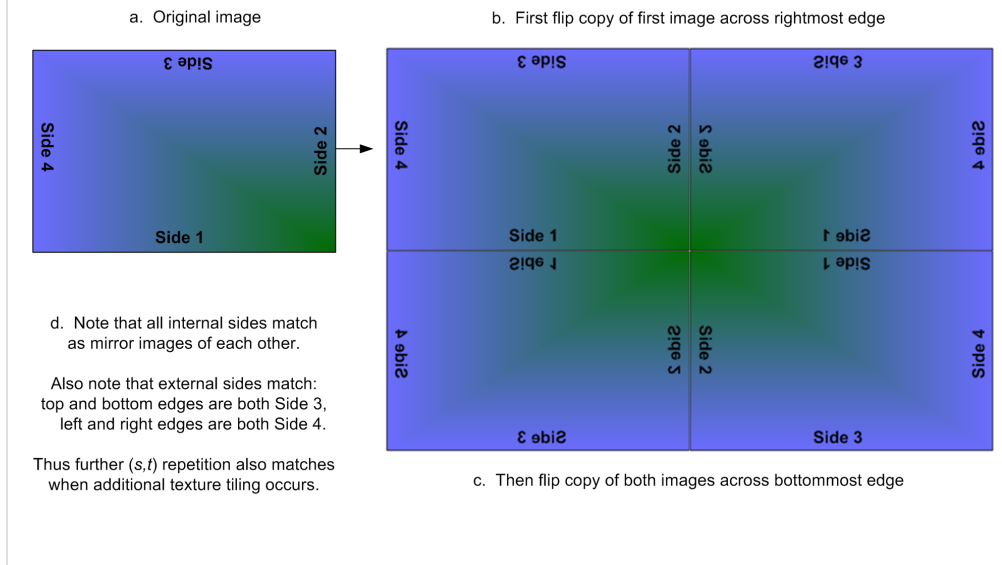
“Figure 18.1 — Texture map coordinate system”
used with permission from X3D Specification

Common fields for texture nodes

repeatS and *repeatT*

- These fields indicate whether the texture image is repeated along a given axis once used
- Default is to use once along each axis, mapping the texture image once from coordinates (0,0) to (1,1)

Texture flipping for (s,t) tile repetition



This is a nice trick for repetitive surfaces such as grass, water, sky, etc. that will show sharp, distracting edge artifacts if simply tiled as they originally appear.

Most image editors are capable of copying, flipping and aligning the quadrant images.

This is not a sufficient technique for smooth repetitive texturing if there are large color differences among the pixels within the original image being tiled.

Image file manipulation tools

Many tools are available for manipulating images, sometimes provided with the operating system

- Adobe Photoshop
- Microsoft Visio, Paint

One of best is free, open source, recommended:

- Gnu Image Manipulation Program (GIMP)
<http://www.gimp.org>

Drawing tools can also be helpful

- OpenOffice Draw, Impress
<http://www.openOffice.org>



ImageTexture node

ImageTexture retrieves a 2D image file and applies it as a texture to geometry

- Commonly used technique, important to master *url* described in Chapter 4 Grouping Nodes
- as part of Inline and Anchor
- Recall that the url field is an ordered list which can include both local (relative) and online addresses to image files
- Might preferentially load online version first, perhaps if it can be updated, and keep a local url value for a backup image



Examples of online imagery updates might be weather-related sky snapshots, webcam views, or perhaps other photography.

Further guidance on url links provided in X3D Scene Authoring Hints, provided within X3D-Edit help system or online at

<http://www.web3d.org/x3d/content/examples/X3dSceneAuthoringHints.html#urls>

ImageTexture file formats

Supported, required image file formats:

- Joint Photographic Expert Group (.jpg) which is good for photographic images
- Portable Network Graphics (.png) which is good for bit-mapped drawings and other images
- Both formats are royalty free, commonly used in Web

Also recommended (but not required)

- Graphics Image Format (.gif), has license restrictions

Other image formats are also allowed

- but support by X3D browser not guaranteed



Specialty image file formats are allowed. There are no restrictions on what file formats can be referenced within the *url* values in an X3D scene.


One common approach to the use of specialty (perhaps high resolution) file formats is to list these first in the url ordered-list array, followed by an alternate version of the image file encoded in a required format such as .png.

In that way, a higher-capability X3D browser will preferentially load the specialty image format, but other regular X3D browsers will skip the unsupported format and then load the fallback url listing the required format.

ImageTexture and Material

It is good practice to accompany ImageTexture with a Material node

- Material is rendered first if network delays are encountered when loading the image file
- Carefully chosen Material *diffuseColor* can reduce sudden color changes when a delayed image file is finally applied
- Underlying Material values are further important and will show through if the texture image includes transparent pixels

 ImageTexture	<p>ImageTexture maps a 2D-image file onto a geometric shape. Texture maps have a 2D coordinate system (s, t) horizontal and vertical, with (s, t) values in range [0.0, 1.0] for opposite corners of the image.</p> <p>Hint: insert Shape and Appearance nodes before adding texture.</p> <p>Warning: bright Material emissiveColor values can wash out some textures.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
url	<p>[url: accessType inputOutput, type MFString CDATA #IMPLIED]</p> <p>Location and filename of image. Multiple locations are more reliable, and Web locations let e-mail attachments work.</p> <p>Hint: Strings can have multiple values, so separate each string by quote marks ["http://www.url1.org" "http://www.url2.org" "etc."].</p> <p>Hint: XML encoding for " is &quot; (a character entity).</p> <p>Warning: strictly match directory and filename capitalization for http links!</p> <p>Hint: can replace embedded blank(s) in url queries with %20 for each blank character.</p>
repeatS	<p>[repeatS: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Horizontally repeat texture along S axis.</p>
repeatT	<p>[repeatT: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Vertically repeat texture along T axis.</p>
containerField	<p>[containerField: NMTOKEN "texture"]</p> <p>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>[class CDATA #IMPLIED]</p> <p>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>

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

MovieTexture node

MovieTexture applies video imagery to geometry

- Same considerations for *url* field as ImageTexture

Usually applied sparingly, because

- movie files are often quite large
- Applying high-frame rate pixels is computationally expensive, which can slow down frame rate and may present low-quality results in some browsers

3D wall or billboard helps make movies viewable

- Important to provide a clear Viewpoint to see it
- Authors might prefer for movie to instead play within Web browser outside X3D scene



MovieTexture fields 1

- *speed* is a rate factor to speed up or slow down movie playback, can be negative to go in reverse
- *startTime* and *stopTime* are used as input controls to begin and end play, usually by routing an SFTIME event from a TimeSensor or TouchSensor
- *pauseTime* and *resumeTime* operate similarly, allowing the movie to pause/resume at same point in time (rather than starting over from beginning)
- *isActive* and *isPaused* are boolean output events that are sent by the MovieTexture node: true when the condition occurs, false when it ends



Events are described further in Chapter 7

MovieTexture fields 1


- *duration_changed* is length of time in seconds for one cycle of the movie
- *elapsedTime* is SFTIME output event sent continuously as movie is playing, cumulatively in seconds without counting any pause durations

Can use LoadSensor (chapter 12) to detect when movie is fully loaded

DEF and USE are important for multiple copies

- Minimize download file size, bandwidth, and delay



 MovieTexture	<p>MovieTexture applies a 2D movie image to surface geometry, or provides audio for a Sound node. First define as texture, then USE as Sound source to see it/hear it/save memory. Texture maps have a 2D coordinate system (s, t) horizontal and vertical, with (s, t) values in range [0.0, 1.0] for opposite corners of the image.</p> <p>Hint: insert Shape and Appearance nodes before adding texture.</p> <p>Hint: provide a viewpoint that allows a clear view of a MovieTexture so that users can easily see all details.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referenceable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USING other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
url	<p>[url: accessType inputOutput, type MFString CDATA #IMPLIED]</p> <p>Location and filename of image Multiple locations are more reliable, and Web locations let e-mail attachments work.</p> <p>Hint: Strings can have multiple values, so separate each string by quote marks ["http://www.url1.org" "http://www.url2.org" "etc."].</p> <p>Hint: XML encoding for " is &quot; (a character entity).</p> <p>Warning: strictly match directory and filename capitalization for http links!</p> <p>Hint: can replace embedded blank(s) in url queries with %20 for each blank character.</p>
loop	<p>[loop: accessType inputOutput, type SBool (true/false) "false"]</p> <p>Repeat indefinitely when loop=true, repeat only once when loop=false.</p>
speed	<p>[speed: accessType inputOutput, type SFloat CDATA "1.0"]</p> <p>Factor for how fast the movie (or soundtrack) is played.</p>
startTime	<p>[startTime: accessType inputOutput, type STime CDATA "0"]</p> <p>Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT.</p> <p>Hint: usually receives a ROUTED time value.</p>
stopTime	<p>[stopTime: accessType inputOutput, type STime CDATA "0"]</p> <p>Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT.</p> <p>Hint: usually receives a ROUTED time value.</p>
repeatS	<p>[repeatS: accessType initializeOnly, type SBool (true/false) "true"]</p> <p>Horizontally repeat texture along S axis.</p>
repeatT	<p>[repeatT: accessType initializeOnly, type SBool (true/false) "true"]</p> <p>Vertically repeat texture along T axis.</p>
duration_changed	<p>[duration_changed: accessType outputOnly, type STime CDATA #FIXED ""]</p> <p>Length of time in seconds for one cycle of movie.</p>

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

isActive	[isActive: outputOnly SBoolLabel; #FIXED ""] isActive true/false events are sent when playback starts/stops.
isPaused	[isPaused: accessType outputOnly, type SBool (true/false) #FIXED ""] isPaused true/false events are sent when MovieTexture is paused/resumed. Warning: not supported in VRML97.
pauseTime	[pauseTime: accessType inputOutput, type STime CData "0"] When time now >= pauseTime, isPaused becomes true and MovieTexture becomes paused. Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. Hint: usually receives a ROUTED time value. Warning: not supported in VRML97.
resumeTime	[resumeTime: accessType inputOutput, type STime CData "0"] When resumeTime becomes <= time now, isPaused becomes false and MovieTexture becomes active. Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. Hint: usually receives a ROUTED time value. Warning: not supported in VRML97.
elapsedTime	[elapsedTime: accessType outputOnly, type STime CData #FIXED ""] Current elapsed time since MovieTexture activated/running, cumulative in seconds, and not counting any paused time. Warning: not supported in VRML97.
containerField	[containerField: NMTOKEN "texture"] 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#MovieTexture>

PixelFormat node

PixelFormat contains the bit pattern of an image

- Written out as set of numeric data within the node

This allows single X3D scene to embed imagery

- Which helps when delivering a self-sufficient scene
- However may increase overall file size

Numeric image data is encoded pixel by pixel,
using a special data type: SFImage

SFImage data type

First three data values:

- Number of width pixels in image
- Number of height pixels in image
- Number of component values in each pixel (0-4)

Appropriate number of pixel values follow






Component values

- 0 for no image, `<ImageTexture image='0 0 0' />`
- 1 for black-white intensity
- 2 for black-white intensity, transparency
- 3 for red-green-blue colors
- 4 for red-green-blue colors, transparency



Pixel values can be in decimal or hexadecimal formats

SFImage examples

Components	SFImage Value	Description	Image
0	0 0 0	Empty image	
1	1 2 1, 0xFF 0x00	Intensity (black & white) example: checkerboard pattern	
2	2 1 2, 0xCCFF 0x2277	Intensity & transparency example	
3	2 4 3, 0xFF0000 0xFF00 0 0 0 0 0xFFFF 0xFFFF00	Red-green-blue (RGB) example	
4	3 2 4, 1 0 0 255, 0 1 0 255, 0 0 1 255, 1 0 0 127, 0 1 0 127, 0 0 1 127	Red-green-blue-alpha (RGBA) example	

X3D for Web Authors, Table 5.18, p.145

Hexadecimal is base sixteen, which is more concise and suitable for binary data. Both decimal (base 10) and hexadecimal data appear in the above example.

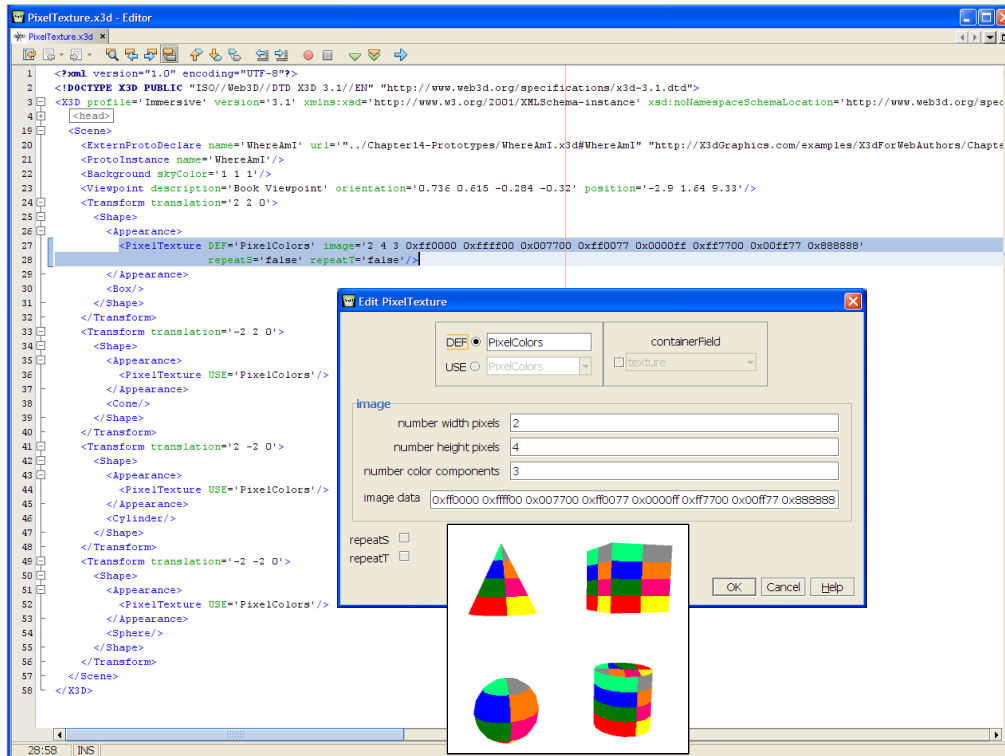
Base 10 digits: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Base 16 digits: 0 1 2 3 4 5 6 7 8 9 a b c d e f

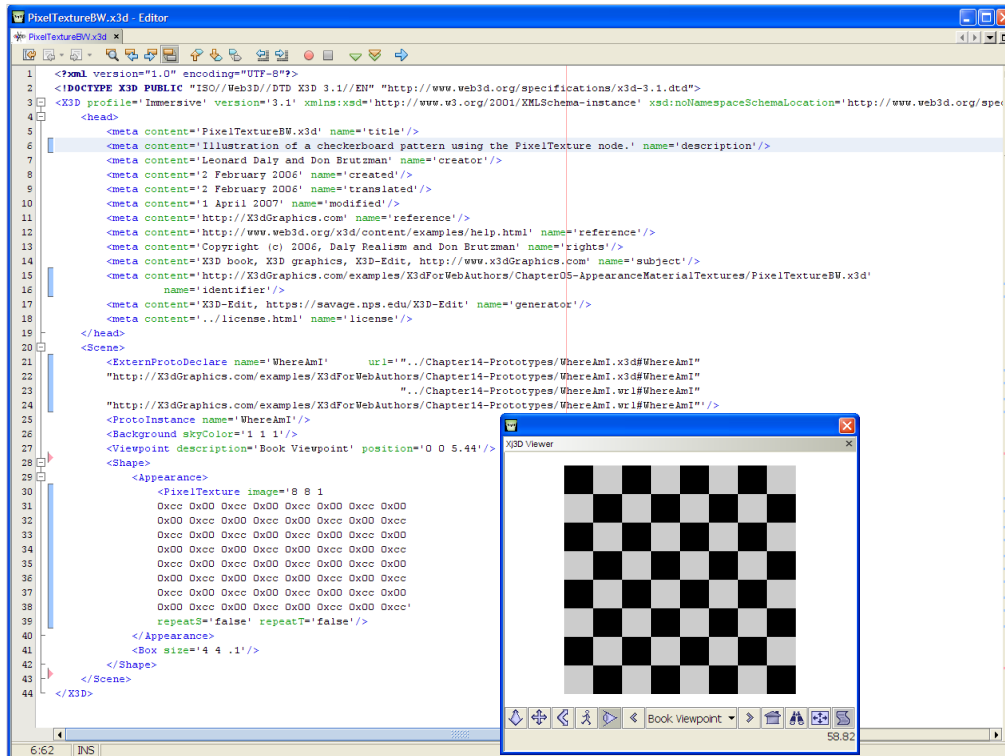
Note X3D/ClassicVRML encoding prefix for hexadecimal data is '0x'

Hexadecimal data is unsigned, sign information is carried in most-significant bit.


binary	octal	decimal	hex	binary	octal	decimal	hex
0000.....	0.....	0.....	0.....	1000.....	10.....	8.....	8
0001.....	1.....	1.....	1.....	1001.....	11.....	9.....	9
0010.....	2.....	2.....	2.....	1010.....	12.....	10.....	a
0011.....	3.....	3.....	3.....	1011.....	13.....	11.....	b
0100.....	4.....	4.....	4.....	1100.....	14.....	12.....	c
0101.....	5.....	5.....	5.....	1101.....	15.....	13.....	d
0110.....	6.....	6.....	6.....	1110.....	16.....	14.....	e
0111.....	7.....	7.....	7.....	1111.....	17.....	15.....	f



<http://x3dgraphics.com/examples/X3dForWebAuthors/Chapter05-AppearanceMaterialTextures/PixelTexture.x3d>



<http://x3dgraphics.com/examples/X3dForWebAuthors/Chapter05-AppearanceMaterialTextures/PixelTextureBW.x3d>

 PixelTexture	<p>PixelTexture creates a 2D-image texture map using a numeric array of pixel values. Texture maps have a 2D coordinate system (s, t) horizontal and vertical, with (s, t) values in range [0.0, 1.0] for opposite corners of the image. Hint: this is a good way to bundle image(s) into a single scene file, avoiding multiple downloads.</p> <p>Warning: aggregate file size can grow dramatically.</p> <p>Hint: insert Shape and Appearance nodes before adding texture.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
image	<p>[image: accessType inputOutput, type SFImage CDATA "0 0 0"]</p> <p>Defines image: width height number_of_components pixel_values. width and height are pixel count, number_of_components = 1 (intensity), 2 (intensity alpha), 3 (red green blue), 4 (red green blue alpha-transparency). intensity example: [1 2 1 0xFF 0x00] intensity-alpha example: [2 2 1 0 255 255 0] red-green-blue example: [2 4 3 0xFF0000 0xFF00 0 0 0 0xFFFF 0xFFFF 0] red-green-blue-alpha example: [needed]</p>
repeatS	<p>[repeatS: accessType initializeOnly, type SFBool (true/false) "true"]</p> <p>Horizontally repeat texture along S axis.</p>
repeatT	<p>[repeatT: accessType initializeOnly, type SFBool (true/false) "true"]</p> <p>Vertically repeat texture along T axis.</p>
containerField	<p>[containerField: NMTOKEN "texture"]</p> <p>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>[class CDATA #IMPLIED]</p> <p>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>

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

TextureTransform node

TextureTransform defines a 2D coordinate transformation for corresponding texture node, to better align images placed on geometry

- 2D translation left/right/up/down
- rotation angle about center
- 2D scaling, uniform or non-uniform

Transformation order remains significant

- translation, rotation, scale (same as Transform)
- However it is applied against coordinate system, **not** image file, so directions are counterintuitive

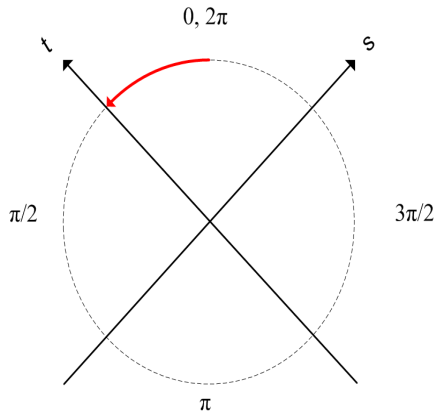
TextureTransform fields

Transformation are (s,t) axes-centric, not image centric, so direction differs from expectations

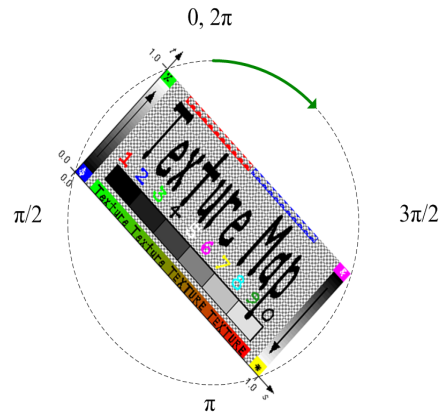
- *translation* controls lateral shift of image file along the polygonal surface, defined using (s,t) values
- *center* and *rotation* modify texture orientation: each makes a change in coordinate system, so the textured image rotates in opposite direction
 - *center* defined using (s,t) values
 - *rotation* defined using radians
- scale similarly opposite: `scale='3 0.5'` shows only 1/3 of texture along s axis, doubled along t axis

TextureTransform rotation

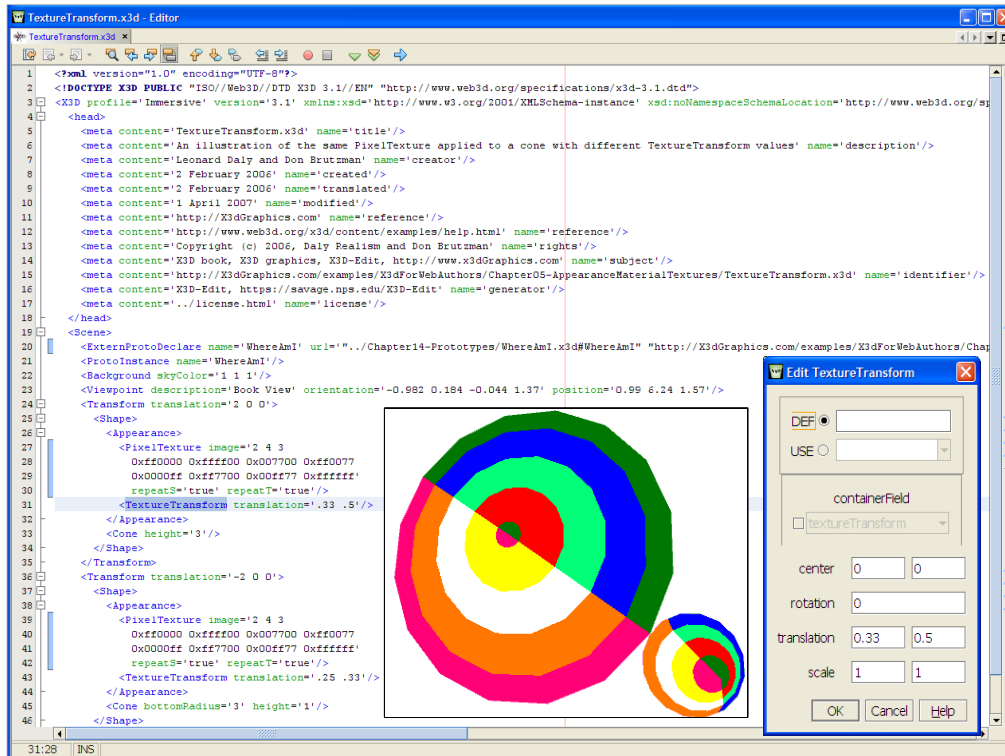
ccw rotation of
geometry texture coordinate axes




opposite rotation of
applied texture images



`<TextureTransform rotation='0.78' />`



<http://x3dgraphics.com/examples/X3dForWebAuthors/Chapter05-AppearanceMaterialTextures/TextureTransform.x3d>

 TextureTransform	TextureTransform shifts 2D texture coordinates to position, orient and scale image patches. Visible effects appear reversed because image changes occur before mapping to geometry Order: translation, rotation about center, non-uniform scale about center. Hint: insert Shape and Appearance nodes before adding TextureTransform.
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: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
translation	[translation: accessType inputOutput, type SFVec2f CDATA "0 0"] Lateral/vertical shift in 2D (s,t) texture coordinates (opposite effect appears on geometry).
center	[center: accessType inputOutput, type SFVec2f CDATA "0 0"] center point in 2D (s,t) texture coordinates for rotation and scaling.
rotation	[rotation: accessType inputOutput, type SFFloat CDATA "0"] single rotation angle of texture about center (opposite effect appears on geometry). Warning: use a single radian angle value, not a 4-tuple Rotation.
scale	[scale: accessType inputOutput, type SFVec2f CDATA "1 1"] Non-uniform planar scaling of texture about center (opposite effect appears on geometry).
containerField	[containerField: NMTOKEN "textureTransform"] 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#TextureTransform>

TextureCoordinate node

TextureCoordinate specifies a set of 2D texture coordinates used by vertex-based nodes


- Such as IndexedFaceSet and ElevationGrid, which are covered in Chapter 6

TextureCoordinate *point* field has (s,t) values corresponding to vertices in parent geometry

- Type MFVec2f, multiple field array of 2-tuple floats
- Default is empty array, corresponds to regular (s,t) values ranging $(0,1)$

Best approach: use special authoring tools



 TextureCoordinate	TextureCoordinate specifies 2D (s,t) texture-coordinate points, used by vertex-based geometry (ElevationGrid, IndexedFaceSet) to map textures to vertices (and patches to polygons). Hint: add Shape and then polygonal/planar geometry before adding TextureCoordinate.
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: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
point	[point: accessType inputOutput, type MFVec2f CDATA #IMPLIED] pairs of 2D (s,t) texture coordinates, either in range [0..1] or higher if repeating.
containerField	[containerField: NMTOKEN "texCoord"] 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#TextureCoordinate>

TextureCoordinateGenerator node

TextureCoordinateGenerator enables the automatic computation and generation of texture coordinates for geometric shapes

- Can serve as substitute for TextureCoordinate node

Eleven procedural modes are provided

- *mode* field, following table explains possible values
- Associated *parameter* field provides setup values

This node is quite complicated and likely requires special authoring-tool support

- May also find support in 3D acceleration hardware



TextureCoordinateGenerator *mode* enumerations and *parameter* values

Mode	Description
SPHERE	Creates texture coordinates for a spherical environment or "chrome" mapping based on the vertex normals transformed to camera space. $u = N_x/2 + 0.5$ $v = N_y/2 + 0.5$ where u and v are the texture coordinates being computed, and N_x and N_y are the x and y components of the camera-space vertex normal. If the normal has a positive x component, the normal points to the right, and the u coordinate is adjusted to address the texture appropriately. Likewise for the v coordinate: positive y indicates that the normal points up. The opposite is of course true for negative values in each component. If the normal points directly at the camera, the resulting coordinates should receive no distortion. The $+0.5$ bias to both coordinates places the point of zero-distortion at the center of the sphere map, and a vertex normal of $(0, 0, z)$ addresses this point. Note that this formula doesn't take account for the z component of the normal.
CAMERASPACE NORMAL	Use the vertex normal, transformed to camera space, as input texture coordinates, resulting coordinates are in -1 to 1 range.
CAMERASPACE POSITION	Use the vertex position, transformed to camera space, as input texture coordinates
CAMERASPACE REFLECTIONVECTOR	Use the reflection vector, transformed to camera space, as input texture coordinates. The reflection vector is computed from the input vertex position and normal vector. $R = 2 \times \text{DotProd}(E, N) \times N - E$; In the preceding formula, R is the reflection vector being computed, E is the normalized position-to-eye vector, and N is the camera-space vertex normal. Resulting coordinates are in -1 to 1 range.
SPHERE-LOCAL	Sphere mapping but in local coordinates
COORD	Use vertex coordinates
COORD-EYE	Use vertex coordinates transformed to camera space
NOISE	Computed by applying Perlin solid noise function on vertex coordinates, parameter contains scale and translation [scale.x scale.y scale.z translation.x translation.y translation.z]
NOISE-EYE	Same as above but transform vertex coordinates to camera space first
SPHERE-REFLECT	Same as above but transform vertex coordinates to camera space first
SPHERE-REFLECT-LOCAL	Similar to "SPHERE-REFLECT", parameter[0] contains index of refraction, parameter[1 to 3] the eye point in local coordinates. By animating parameter [1 to 3] the reflection changes with respect to the point. Resulting coordinates are in -1 to 1 range.

X3D for Web Authors, Table 5.25, p.154

<div> <div> <div>u</div> <div>v</div> <div>u</div> <div>v</div> </div> <div>TextureCoordinateGenerator</div> </div>	<p>TextureCoordinateGenerator computes 2D (s,t) texture-coordinate points, used by vertex-based geometry (ElevationGrid, IndexedFaceSet) to map textures to vertices (and patches to polygons).</p> <p>Hint: add Shape and then polygonal/planar geometry before adding TextureCoordinateGenerator.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
mode	<p>[mode: accessType inputOutput, (SPHERE CAMERASPACE NORMAL CAMERASPACE POSITION CAMERASPACE REFLECTION VECTOR SPHERE-LOCAL COORD COORD-EYE NOISE NOISE-EYE SPHERE-REFLECT SPHERE-REFLECT-LOCAL) "SPHERE"]</p>
parameter	<p>[parameter: accessType inputOutput, type MFVec2f CDATA #IMPLIED]</p>
containerField	<p>[containerField: NMTOKEN "texCoord"]</p> <p>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>[class CDATA #IMPLIED]</p> <p>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>

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

[back to Table of Contents](#)

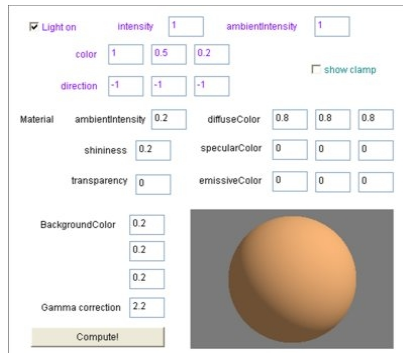
Additional Resources



Additional Resources

Pellucid materials editor

- Eric Haines, copyright (c) 1997
- <http://tog.acm.org/resources/applets/vrml/pellucid.html>



web|3D
CONSORTIUM



[back to Table of Contents](#)

Chapter Summary



Chapter Summary

Appearance affects associated geometry, containing the following fields

Visual surface properties that interact with lights

- Material and TwoSidedMaterial
- LineProperties and FillProperties

Texture nodes wrap images onto geometry

- ImageTexture, MovieTexture, PixelTexture and MultiTexture
- TextureTransform, TextureCoordinate and TextureCoordinateGenerator



[back to Table of Contents](#)

References



References 1

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

- Chapter 5, Appearance, Material and Textures
- <http://x3dGraphics.com>
- <http://x3dgraphics.com/examples/X3dForWebAuthors>

X3D Examples Help

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



References 2

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 10 – Materials
- Chapter 17 – Textures
- Chapter 18 – Texture Mapping
- Chapter 21 – Shiny Materials



Contact

Don Brutzman

brutzman@nps.edu

<http://web.nps.navy.mil/~brutzman>

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

1.831.656.2149 voice

1.831.656.7599 fax



Open-source license

Copyright (c) 1995-2008 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.