

CyberX3D for Java User's Guide

Document Revision History

Modified	Description
June 19, 2002	Added a description of new constructor of VRML97Loader.
April 5, 2002	Added a description of PROTO option for loading.
March 3, 2002	This was the first release of the User's Guide.

Table of Contents

1.Introduction

CyberVRML97 for Java is a development package for VRML97/2.0 and Java3D programmers. Using the package, you can easily read and write the VRML files, set and get the scene graph information, draw the geometries, run the behaviors easily.

2.Setup

To use the package, copy the package into your JDK and JRE library directory. For example,

```
copy cv97r???.jar C:\jdk1.x\jre\lib\ext\  
copy cv97r???.jar C:\Program Files\JavaSoft\JRE\1.x\lib\ext\
```

Otherwise add the package into your CLASSPATH environmental variable. For example,

```
set CLASSPATH=.;c:\src\java\cv97r???.jar
```

Scene Graph

In the CyberVRML97, the scene graph is correction and hierarchical arrangement of VRML nodes.

There are two ways to build a scene graph dynamically, you can load it from a VRML file, or you can create new VRML node and add the node into the scene graph.

Loading Scene Graph

Use `SceneGraph::load()` to load a scene graph from a VRML file. The `load()` clears all VRML nodes in the current scene graph.

```
SceneGraph sg = new SceneGraph();
sg.load("world.wrl");
```

If your scene graph has some VRML nodes and you want to add a new scene graph from a VRML file into the current scene graph, use `SceneGraph::add()`.

```
SceneGraph sg = new SceneGraph();
.....
sg.add("world.wrl");
```

If your scene graph has some VRML nodes and you want to add a new scene graph from a VRML file into the current scene graph, use `SceneGraph::add()`.

```
SceneGraph *sceneGraph = new SceneGraph();
.....
sceneGraph->add("world.wrl");
```

If `SceneGraph::load()` or `SceneGraph::add()` can't read your VRML file normally, the methods return false. To know the error in more detail, use `SceneGraph::getParserErrorMessage()`. The following example shows the parser error when the loading is failed.

```
SceneGraph sg = new SceneGraph();
boolean result = sg.load("world.wrl");
if (result == false) {
    String errMsg = sg.getParserErrorMessage();
    System.out.println(errMsg);
}
```

The CyberVRML97 supports PROTO defines of VRML97 using the preprocessor, but the preprocessor is inactive by default. If you want to use the preprocessor, use `SceneGraph::setOption()` with `USE_PREPROCESSOR` before loading. For example,

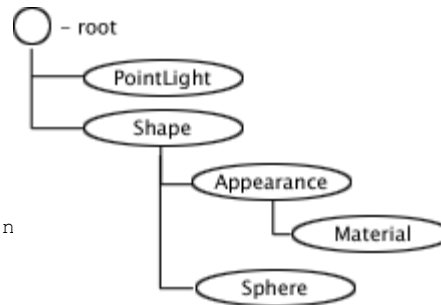
```
SceneGraph sg = new SceneGraph();
sg.setOption(SceneGraph.USE_PREPROCESSOR);
sg.load("world.wrl");
```

Building Scene Graph

The CyberVRML97 has all C++ classes of VRML nodes, you can add and remove the all nodes dynamically. Use `SceneGraph::addNode()` to add a new node as a root node of the scene graph. Use `Node::addChildNode()` to add a new node as a child node of the other node.

The following example adds a `PointLight` and `Shape` node that has an appearance and a geometry node into an empty scene graph.

```
SceneGraph sg;
// Add a PointLight node
PointLightNode plight = new PointLightNode();
// Add a Shape node as a root node
ShapeNode shape = new ShapeNode();
sg.addNode(shape);
// Add an Appearance node as a child node of the Shape n
Appearancenode app = new AppearanceNode();
shape.addChildNode(app);
// Add a Material node as a child node of the Appearance node
MaterialNode mat = new MaterialNode();
mat.setDiffuseColor(1.0f, 0.0f, 0.0f); // Red
app.addChildNode(mat);
// Add a Sphere node as a child node of the Shape node
SphereNode sphere = new SphereNode();
shape.addChildNode(sphere);
sphere.setRadius(10.0f);
```



Use `Node::remove()` to remove a node from the current scene graph. The following example uses `Node::remove()` to remove a `PointLight` node from a loaded scene graph.

```
SceneGraph sg;
sg.load("world.wrl");
// Remove first PointLight node
PointLightNode plight = sg.findPointLightNode();
if (plight != null)
    plight.remove();
```

Scene Graph Output

Use `SceneGraph::save()` to save a current scene graph into a VRML file.

```
sceneGraph.save("newworld.wrl");
```

Use `SceneGraph::print()` to output a current scene graph into a default console.

```
sceneGraph.print();
```

Scene Graph Traversal

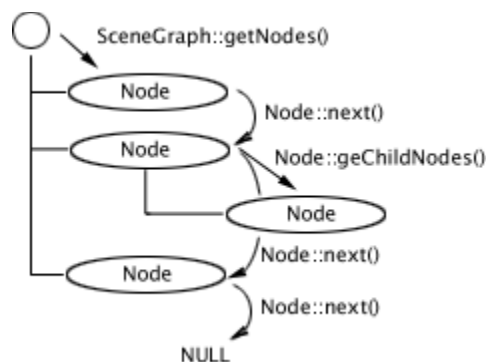
The scene graph has the VRML information as a collection of Node class instances. The Node class is a super class of all VRML node classes of CyberVRML97. There are two ways to access the nodes.

The first way is to use SceneGraph::getNodes() with Node::next() and Node::getChildNodes(). For example, if you want to get all viewpoint nodes in a scene graph

```
void GetViewpointInfomation(Node node)
{
    if (node.isViewpointNode()) {
        ViewpointNode view = (Viewpoint)node;
        // Get a viewpoint information
        .....
    }
    for (Node cnode=node.getChildNodes(); cnode; cnode=cnode.next())
        GetViewpointInfomation(cnode);
}

void main()
{
    .....
    SceneGraph sg = new SceneGraph();
    sceneGraph.load("world.wrl");
    for (Node node=sg.getNodes(); node; node=node.next())
        GetViewpointInfomation(node);
    .....
}
```

SceneGraph::getNodes() returns a first node that are added into the scene graph root. Node::next() returns a next node in the same hierarchy, Node::getChildNodes() returns a first child node that are added into the parent node. The methods returns null if the next node does not exist.



The other way is to use SceneGraph::getNodes()with Node::nextTraversal(). The way is handier than the first one. For example, if you want to get all viewpoint nodes in the scene graph

```
void main()
{
    .....
    SceneGraph sg = new SceneGraph();
```

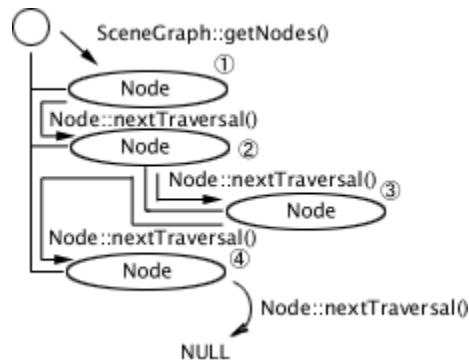


```

sg.load("world.wrl");
for (Node node=sg.getNodes(); node; node=node.nextTraversal())
    if (node.isViewpoint()) {
        ViewpointNode view = (ViewpointNode)node;
        // Get a viewpoint information
        .....
    }
}
.....
}

```

Node::nextTraversal() is similar to Node::next(), but Node::nextTraversal tries to get a next node from the parent node when the next node does not exist. This Node::nextTraversal() is overridden in the sub classes, too.



If you want to get only same type nodes, use SceneGraph::find<nodetype>Node() instead of SceneGraph::getNodes() with Node:: nextTraversalSameType() that returns a next same class node. For example, if you want to get only all viewpoint nodes in the scene graph

```

SceneGraph sg = new SceneGraph();
sg.load("world.wrl");

for (ViewpointNode view=sg.findViewpointNode(); view;
view=(ViewpointNode)view.nextTraversalSameType())) {
    // Get a viewpoint information
    .....
}

```

Finding Node

Use SceneGraph::findNode() to find a named node by DEF keyword or Node::setName(). The following example loads a VRML file, "world.wrl", and gets a node that is named as "MountFuji".

```

SceneGraph sg;
sg.load("world.wrl")
Node node = sg.findNode("MountFuji");

```

Use SceneGrahp::get<nodetype>Nodes() to get a specified first node from the root hierarchy. For example, you want to get a viewpoint that is a first viewpoint node in the scene graph root, use SceneGraph::getViewpointNodes();

```

SceneGraph sg;
sg.load("world.wrl")

```

```
ViewpointNode defaultView = sg.getViewpointNodes();
```

Node

The CyberVRML97 node name is identical to the VRML node name. For example, you would use the BoxNode class if you want to use the Box node.

Creating Node

The node class has the default constructor only. Use the default constructor to create the new node, and set the field property. The following example creates a sphere and set the property.

```
SphereNode spNode = new SphereNode();
spNode.setRadius(10.0);
```

Node Type

If you want to know the node type, use Node::getType() or Node::is<nodetype>Node();

Node::getType() returns the node type as a string, Node::is<nodetype>Node() returns true when the node is the specified type. For example, you want to know whether a node is ViewpointNode.

```
SceneGraph sg = new SceneGraph();
sg.load("world.wrl");
for (Node node=sg.getNodes(); node; node=node.nextTraversal())
    String nodeType = node.getType();
    if (node.isViewpoint() || nodeType.equals("Viewpoint") == 0)
        System.out.println("This node is ViewpointNode !!");
}
```

Node Name

If you load a VRML file that has some named nodes by DEF keyword, you can get the name using Node::getName(); Use Node::setName() to name a node.

```
SceneGraph sg;
sg.load("world.wrl");
ShapeNode mtNode = sg.getShapeNodes();
mtNode.setName("MtFuji");
```

The named nodes are output using DEF keyword when you save the scene graph.

```
#VRML V2.0 utf8
.....
DEF MtFuji Shape {
    .....
}
.....
```

Accessing Fields

The node class has set<fieldname>() and get<fieldname>() to access to the VRML fields, and has getN<fieldname>s() if the field is multi field, MFString etc.. For example, the AnchorNode class has the following methods.

```
class AnchorNode {
    void      setDescription(String value);
    String    getDescription();
    void      addParameter(String value);
    int       getNParameters();
    String    getParameter(int index);
    void      addUrl(String value);
    int       getNUrls();
    String    getUrl(int index);
    void      setUrl(int index, String urlString);
};
```

The Node classes has only basic field access methods. Use get<fieldname>Field() to access the field in more detail. The method returns the field class itself. You can operate the field in more detail to the field class. The following example gets a color field of a Color Node in an IndexedFaceSet node, and changes all the colors to red.

```
IndexedFaceSetNode idxNode = .....
ColorNode colNode = idxNode.getColorNodes();
MFColor colField = colNode.getColorField();
int colCnt = colField.size();
for (int n=0; n<colCnt; n++)
    colField.set1Value(n, 0xff, 0x00, 0x00) // Red
```

Adding to Scene Graph

Use SceneGraph::addNode() to add the node as a root node of the scene graph. The following example creates a transform node add the node to the scene graph root.

```
SceneGraph sg;
TransformNode transNode = new TransformNode();
sg.addNode(transNode);
```

Adding Child Node

Use Node::addChildNode() to add the node as a child node of the other node. The following example creates a shape and a box, add the shape to the scene graph root, and add the box to the shape.

```
SceneGraph sg;
ShapeNode shapeNode = new ShapeNode();
BoxNode boxNode = new BoxNode();
sg.addNode(shapeNode);
shapeNode.addChildNode(boxNode);
```

Getting Child Node

There are two ways to get child nodes. The first way is to use `Node::getNChildNodes()` that returns a count of child nodes, and `Node::getChileNode()` that returns a selected child node. The following example shows the name and the type of all child nodes.

```
void PrintChildNodes(Node node)
{
    int childNodeCnt = node.getNChildNodes();
    for (int n=0; n<childNodeCnt; n++) {
        Node childNode = node.getChildNode(n);
        char nodeType = childNode.getType();
        char nodeName = childNode.getName();
        System.out.println("[ " + n + " ] = " + nodeType + ", " + nodeName);
    }
}
```

The other way is to use `Node::getChildNodes()` that returns a first child node with `Node::next()` returns a next child node in the same parent node. The `Node::getChildNodes()` and the `Node::next()` returns NULL if the node does not exist. The way is handier and faster than the first one. The following example shows the name and the type of all child nodes.

```
void PrintChildNodes(Node node)
{
    Node childNode = node.getChildNodes();
    while (childNode!= NULL) {
        String nodeType = childNode.getType();
        String nodeName = childNode.getName();
        System.out.println("[ " + n + " ] = " + nodeType + ", " + nodeName);
        childNode = childNode.next();
    }
}
```

Removing from SceneGraph or Parent Node.

Use `Node::remove()` to remove a node from the scene graph root or the parent node. The following example removes a point light from the scene graph.

```
SceneGraph sg;
.....
PointLightNode plight = sg.findPointLightNode();
if (plight != null)
    plight.remove();
```

Instance Node

You can create an instance of a node like USE keyword of VRML97. Use `Node::createInstanceNode()` to create the instance node.

```
SceneGraph sg;
ShapeNode shape = new ShapeNode();
shape.setName("BOX");
BoxNode box = new BoxNode();
box.setSize(10.0, 20.0, 30.0);
sg.addNode(shape);
shape.addChildNode(box);
Node shapeInstance = shape.createInstanceNode();
sg.addNode(shapeInstance);
```

The scene graph is output as the following when you save the scene graph.

```
DEF BOX Shape {
    geometry Box {
        size 10 20 30
    }
}
USE BOX
```

Field

The node has several fields and the field is a property or attribute of a node. The field is identical to the VRML field name. For example, you would use the SFBool class if you want to use the SFBool field.

Use `get<fieldname>Field()` to get the field. The following example gets a color field of a Color Node in an IndexedFaceSet node, and changes all the colors to red.

```
IndexedFaceSetNode idxNode = .....
ColorNode colNode = idxNode.getColorNodes();
MFColor colField = colNode.getColorField();
int colCnt = colField.size();
for (int n=0; n<colCnt; n++)
    colField.set1Value(n, 0xff, 0x00, 0x00) // Red
```

File Formats

The CyberVRML97 can import the following geometry file formats. Use `SceneGraph::load()` to import the geometry files. The CyberVRML97 converts from the imported information into VRML97 nodes and adds the nodes into the scene graph when the file format is not VRML2.0/97.

VRML2.0/97 (*.wrl)

The CyberVRML97 can import all information in a VRML2.0/97 file.

Autodesk 3DS (*.3ds)

The CyberVRML97 imports only the following information from a 3DS file, and ignores the other information.

Chunk ID	Description
0xA010	Material Ambient Color
0xA020	Material Diffuse Color
0xA030	Material Specular Color
0xA040	Material Shininess
0x4100	Triangle Set
0x4110	Triangle Point Set
0x4120	Triangle Face Set

Autodesk DXF (*.dxf)

The CyberVRML97 imports only the polylines and face3Ds information from a DXF file.

Wavefront OBJ (*.obj)

The CyberVRML97 imports only the following information from the specified OBJ file, and ignores the other information. The CyberVRML97 doesn't read the map files and the material files.

Chunk ID	Description
v	Vertex Position
vn	Vertex Normal
f	Face Index

SENSE8 NFF (*.nff)

The CyberVRML97 imports only the vertex positions and the polygon indices with the color from the specified NFF file, and ignores the other information.

STL (*.stl)

The CyberVRML97 supports the ASCII format.

X3D (*.x3d / *.xml)

The CyberVRML97 supports . The CyberVRML97 can import all tag information without the EXTERNPROTO and PROTO defines in the specified X3D file.

Java3D

Rendering / Behavior

The CyberVRML97 can draw shape nodes with the behaviors automatically with Java3D. Using the feature, you can create your original VRML browser with Java3D easily. To use the feature, you have to create an instance of `vrml.j3d.SceneGraphJ3dObject` class with a target `Canvas3D`, and set the instance to the scene graph. For example,

```
public class ViewerJ3D extends Frame implements Constants {
    private Scene Graph sg;
    private SceneGraphJ3dObject sgObject;
    public ViewerJ3D(){
        super("VRML Simple Viewer");
        .....
        sg = new SceneGraph(SceneGraph.NORMAL_GENERATION);
        setLayout(new BorderLayout());
        Canvas3D c = new Canvas3D(null);
        add("Center", c);
        sgObject = new SceneGraphJ3dObject(c, sg);
        sg.setObject(sgObject);
        .....
        setSize(400,400);
        show();
    }
    .....
}
```

The CyberVRML97 supports the following nodes in the current release.

Anchor ^(*1)	Fog	PositionInterpolator
Appearance	Group	ProximitySensor
Background ^(*2)	ImageTexture	ScalarInterpolator
Billboard	IndexedFaceSet	Script
Box	IndexedLineSet	Shape
Collision ^(*3)	Inline	Sphere
Color	LOD	SpotLight
ColorInterpolator	Material	Switch
Cone	Normal	Text
Coordinate	NormalInterpolator	TextureCoordinate
CoordinateInterpolator	OrientationInterpolator	TextureTransform
Cylinder	PixelTexture	TimeSensor
DirectionalLight	PointLight	Transform
ElevationGrid	PointSet	Viewpoint
Extrusion		

*1) Not support the URL field jump, *2) Support only the first skyColor, *3) Not support the collision detecting.

There are two ways to execute the behaviors. If you want to execute the behaviors as background tasks (threads), you should use `SceneGraph::startSimulation()` and `SceneGraph::stopSimulation()`. Otherwise, If you want to execute the

behaviors when you want, you should use `SceneGraph::update()`. The `SceneGraph::update()` executes the behaviors at once. The following example executes execute the behavior as the background tasks.

```
Scene Graph sg = new Scene Graph();
sg.load("world.wrl");
sg.startSimulation();
```

File Loader

The CyberVRML97 supplies some file loader classes for Java3D to load the following file format. The loader classes implements “com.sun.j3d.loaders”.

File Format	Loader Class (cv97.j3d.loader)
VRML2.0/97 (*.wrl)	VRML97Loader.
Autodesk 3DS (*.3ds)	A3DSLoader
Autodesk DXF (*.dxf)	DXFLoader
Wavefront OBJ (*.obj)	OBJLoader
SENSE8 NFF (*.nff)	NFFLoader
STL (*.stl)	STLLoader

The following example loads an OBJ file using the OBJLoader class, and gets the scene group.

```
import cv97.j3d.loader.*;
.....
OBJLoader objLoader = new OBJLoader();
Scene s = objLoader.load(filename);
SceneGroup sgGroup = scene.getSceneGroup();
```

The constructors have no parameter, but you can set a parameter of Canvas3D to the constructor of the VRML97Loader if you want to load images files of ImageTextureNode.

```
Canvas3D c3d = .....
VRML97Loader wrlLoader = new VRML97Loader(c3d);
Scene s = wrlLoader.load(filename);
```

The VRML97Loader in the current release converts the following VRML97 nodes into Java3D nodes.

Anchor (Group)	IndexedLineSet (IndexedLineArray)
Appearance (Appearance)	Inline
Billboard (TransformGroup)	LOD (Switch)
Box (QuadArray)	Material (Material)
Collision (Group)	Normal
Color	PixelTexture (Texture2D)
Cone (TriangleArray)	PointLight (PointLight)
Coordinate	PointSet (IndexedPointArray)
Cylinder (TriangleArray)	Shape (Shape3D)
DirectionalLight (DirectionalLight)	Sphere (TriangleArray)
ElevationGrid (IndexedTriangleArray)	SpotLight (SpotLight)
Extrusion (IndexedTriangleArray)	Switch (Switch)
Fog (ExponentialFog or LinerFog)	Text (Text3D)
Group (Group)	TextureCoordinate
ImageTexture (Texture2D)	TextureTransform (TextureAttributes)

IndexedFaceSet
(IndexedTriangleArray)

Transform (Transform)

File Saver

The CyberVRML97 can export a scene graph of Java3D to a VRML97 file using the VRMLSaver class. The following example exports a BranchGroup of Java3D into a VRML97 file.

```
import cv97.j3d.*;

.....

BranchGroup j3dBranchGroup = .....
VRML97Saver saver = new VRML97Saver();
saver.setBranchGroup(j3dBranchGroup);
saver.save("j3dworld.wrl");
```

The VRML97Saver in the current release exports the following Java3D nodes into the VRML97 nodes.

AmbientLight (PointLight)	PointLight (PointLight)
Appearance (Appearance)	PointArray (PointSet)
Background (Background)	Shape3D (Shape)
Billboard (Billboard)	SpotLight (SpotLight)
BranchGroup (Group)	Switch (Switch)
DirectionalLight (DirectionalLight)	Text3D (Text)
ExponentialFog (Fog)	Texture2D (PixelTexture)
Group (Group)	TransformGroup (Transform) ^(*1)
IndexedTriangleArray (IndexedFaceSet)	TriangleArray (IndexedFaceSet)
IndexedQuadArray (IndexedFaceSet)	TriangleFanArray (IndexedFaceSet)
LineArray (IndexedLiseSet)	TriangleStripArray (IndexedFaceSet)
LineStripArray (IndexedLiseSet)	QuadArray (IndexedFaceSet)
LinearFog (Fog)	View (Viewpoint)
Material (Material)	

*1) Not support all matrix information

License

CyberVRML97 for Java is provided "AS IS". Licenser disclaims all warranties, including but not limited to, all express or implied warranties of merchant ability and fitness for a particular purpose.

Everyone can use the CyberVRML97 for commerce or personal purposes free. However, If you want to distribute your software using the CyberVRML97, you have to add state that "Portions of this software is based in part on the CyberVRML97 package written by Satoshi Konno" into the program or document.