




VRML: Transformations and Texture Mapping

Day 3: VRML Lesson 2

1000 Galliher Dr. | Fairmont, WV 26555-2720

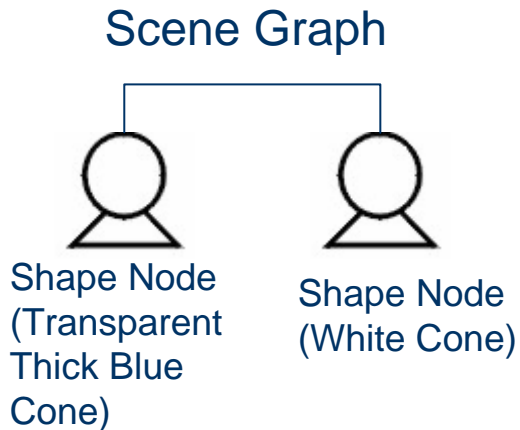
Phone 304.368.9300 | www.wvhtf.org

- Lessons
 - Transformations and Node Hierarchy
 - Translation, Scale, and Rotation
 - Composite Hierarchical Transformation
 - Basic Geometric primitives
 - Using *DEF / USE*
 - Texture Mapping, Backgrounds and Inlining
- Programming Assignment

- A group node that transforms its children nodes
- Scene Graph representation \Rightarrow 
- Important fields of Transform

```
Transform {  
    children    []  
    center     0 0 0  
    scale      1 1 1  
    rotation   0 0 1 0  
    translation 0 0 0  
}
```

- Transparent Blue Cone and White
Cone



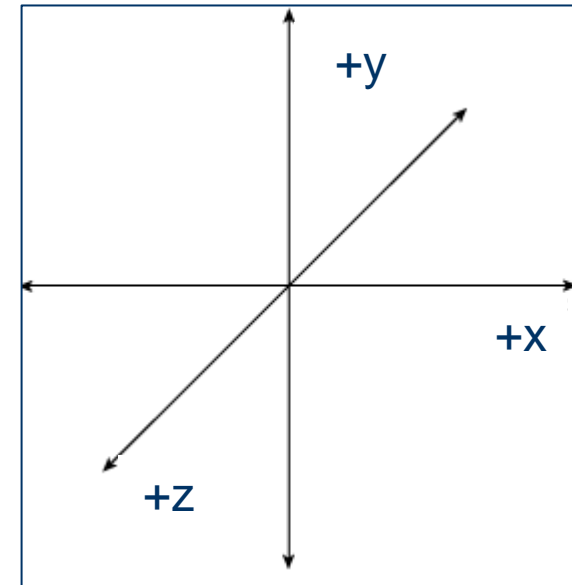
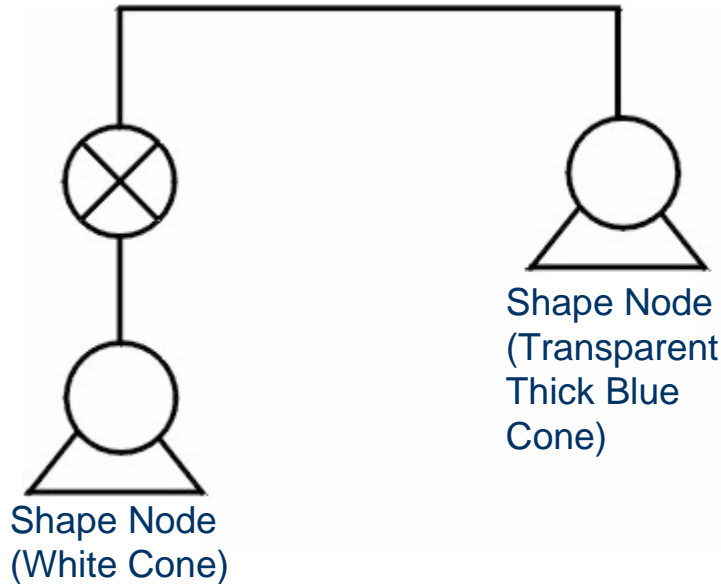
```
#VRML V2.0 utf8

Shape {
  appearance Appearance {
    material Material {
      diffuseColor 0 0 1
      transparency 0.25
    }
  }
  geometry Cone {
    bottomRadius 2
    height 2
  }
}

Shape {
  geometry Cone {}
}
```

Program day3_1_BlueAndWhiteCone.wrl

- Translate, Scale, or Rotate the solid cone

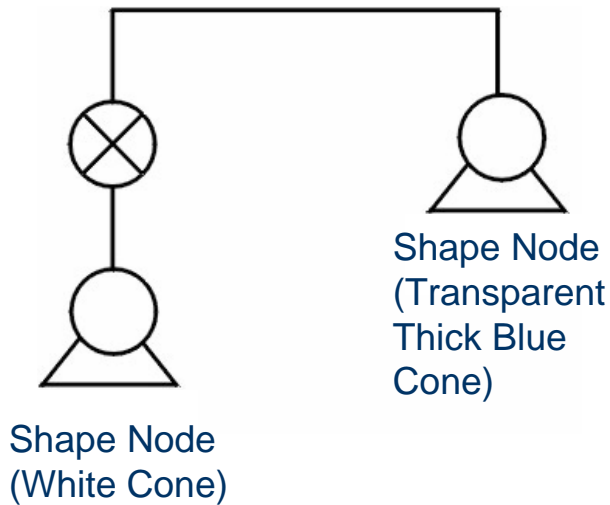


Coordinate Axes

Origin (0,0,0) in
Center of Screen

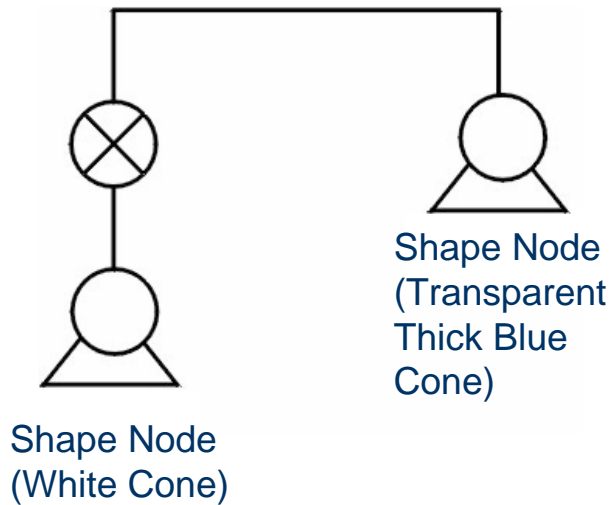
- Use the Transform Node
- Every Node starts at the origin by default

- Translate the Solid White Cone



```
... Blue Cone Code
Transform
{
  translation 1 2 0
  children
  [
    Shape
    {
      geometry Cone
      {
      }
    }
  ]
}
```

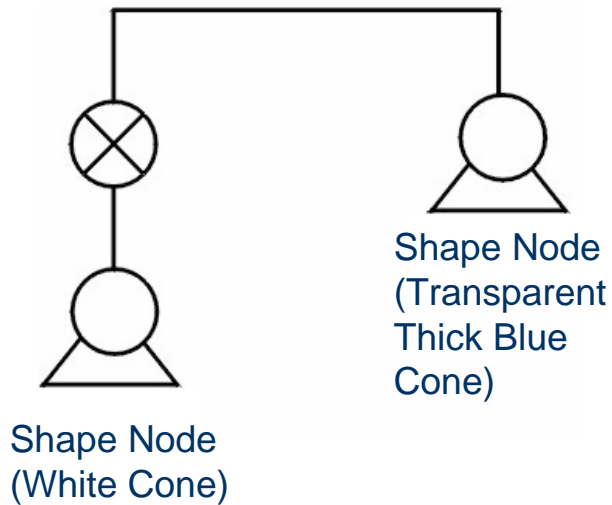
- Scale the Solid White Cone



```
... Blue Cone Code  
Transform  
{  
  # translation 1 2 0  
  scale 3 2 0  
  children  
  [  
    Shape  
    {  
      geometry Cone  
      {  
      }  
    }  
  ]  
}
```

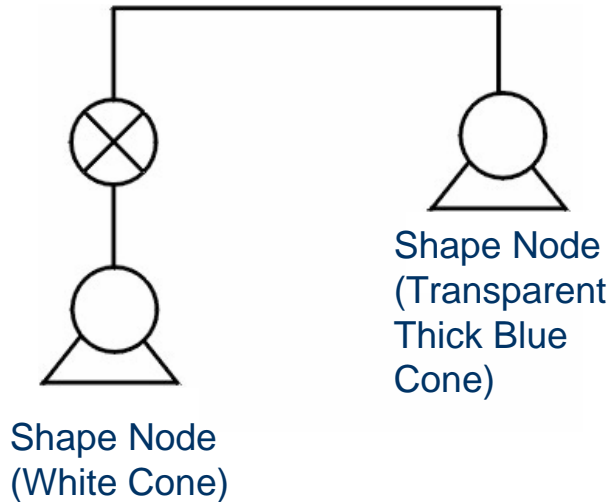
- In Transform:
 - rotation x y z radians
- Radians = Degrees x .017444
 - Examples
 - $45 \times .017444 = .78$
 - $90 \times .017444 = 1.57$
 - $180 \times .017444 = 3.14$
 - $-90 \times .017444 = -1.57$
 - $17 \times .017444 = .30$

- Rotate the Solid White Cone



```
... Blue Cone Code  
Transform  
{  
  # translation 1 2 0  
  # scale 3 2 0  
  rotation 0 0 1 .78 # 45 degrees  
  
  children  
  [  
    Shape  
    {  
      geometry Cone  
      {  
      }  
    }  
  ]  
}
```

- Change the Center of Rotation

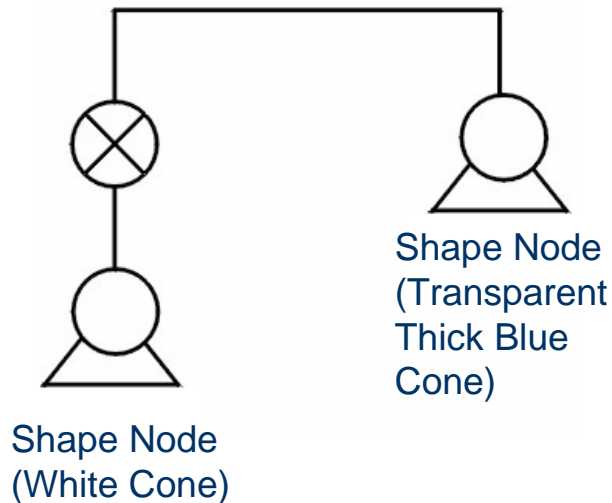


NOTE: Center of Rotation is
at (0,0,0) by default

```
... Blue Cone Code
Transform
{
  # translation 1 2 0
  # scale 3 2 0
  rotation 0 0 1 .78
  center 0 3 0
  children
  [
    Shape
    {
      geometry Cone
      {
      }
    }
  ]
}
```

Lessons: All Together Now except Center

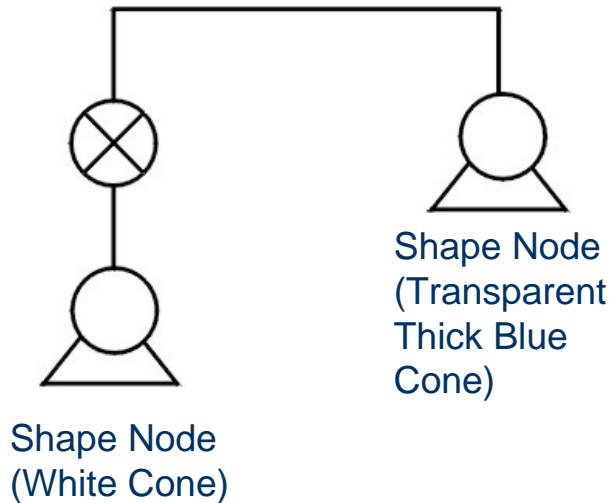
- Change All Transformation Fields



```
... Blue Cone Code  
Transform  
{  
  translation 1 2 0  
  scale 3 2 0  
  rotation 0 0 1 .78  
  # center 0 3 0  
  children  
  [  
    Shape  
    {  
      geometry Cone  
      {  
      }  
    }  
  ]  
}
```

Program day3_6_AllTogetherNow_No_COR.wrl

- Change All Transformation Fields



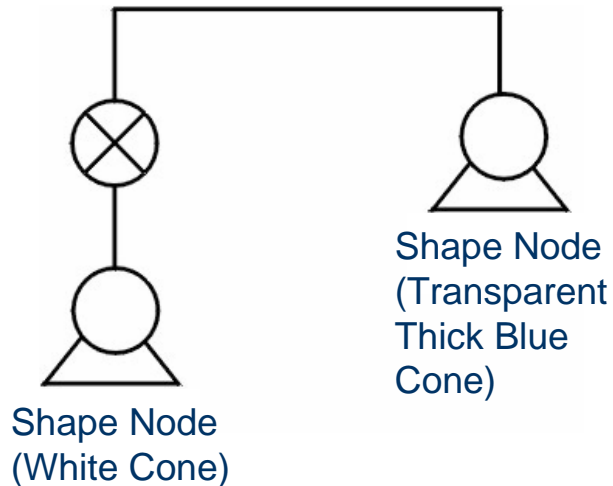
IMPORTANT: Changing Scale and Center of Rotation produces a weird side effect in VRML. (See next slide.)

```

... Blue Cone Code
Transform
{
  translation 1 2 0
  scale 3 2 0
  rotation 0 0 1 .78
  center 0 3 0
  children
  [
    Shape
    {
      geometry Cone
      {
      }
    }
  ]
}

```

- Change Scale and Center of Rotation



IMPORTANT: Changing Scale and Center of Rotation produces a weird side effect of translating object!

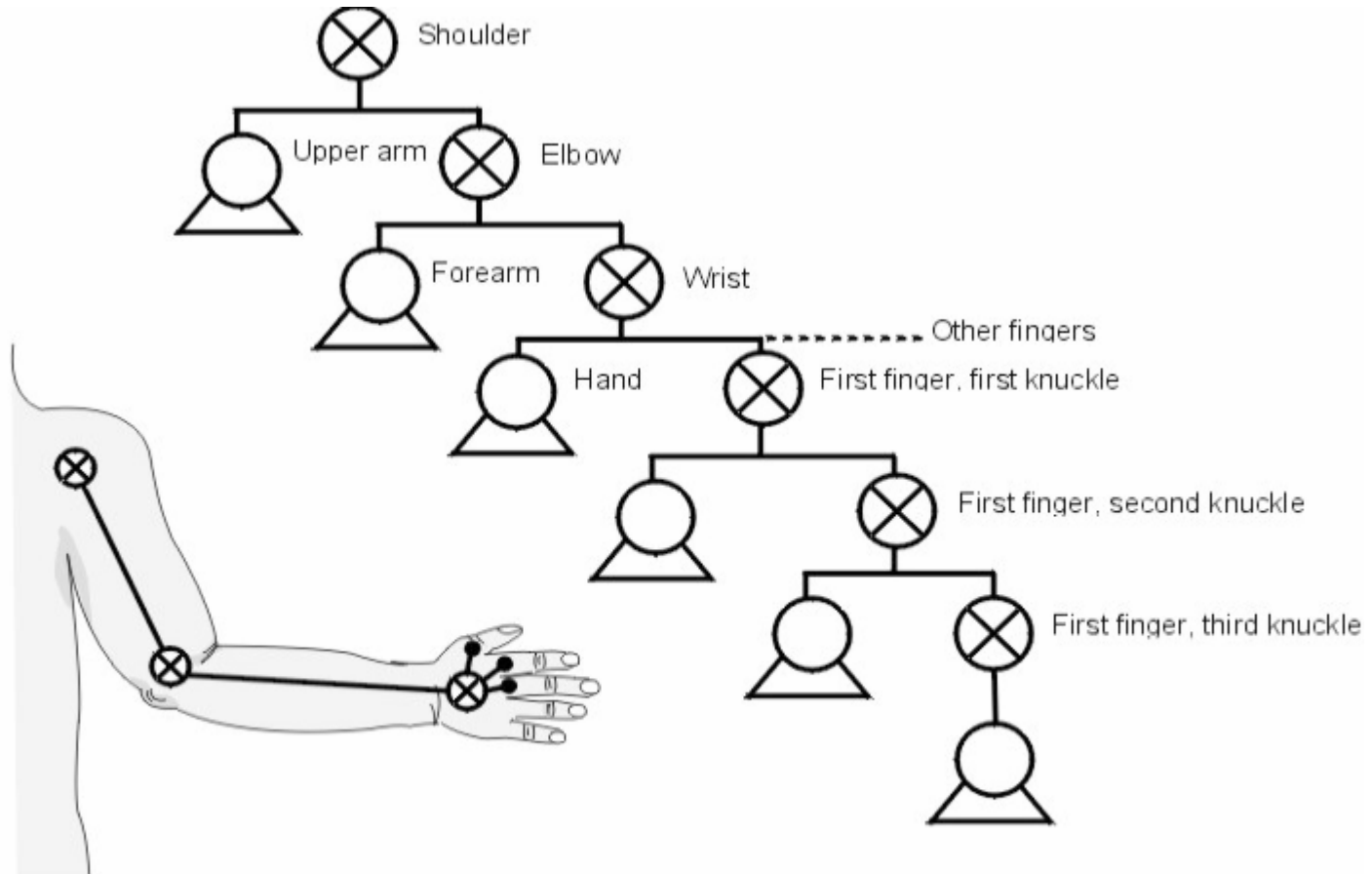
```

... Blue Cone Code
Transform
{
  # translation 1 2 0
  scale 3 2 0
  # rotation 0 0 1 .78
  center 0 3 0
  children
  [
    Shape
    {
      geometry Cone
      {
      }
    }
  ]
}

```

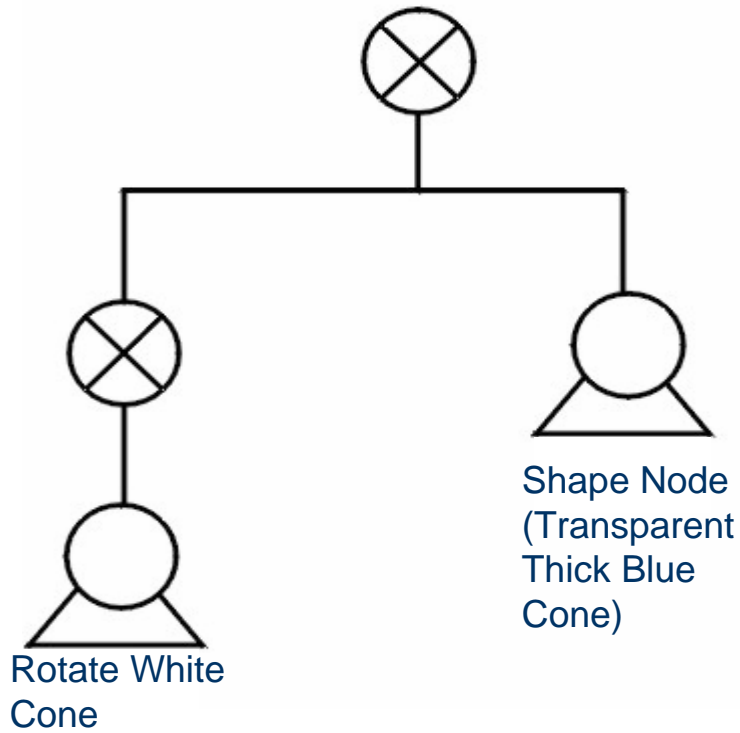
Program day3_8_SideEffect_ScaleAndCenter.wrl

Lessons: Composite hierarchical transformation



Lessons: Simple Composite Example

- White node is rotated first
- Both White and Blue nodes are then translated



```
#VRML V2.0 utf8
```

```
Transform
```

```
{
```

```
  translation 1 2 0
```

```
  children
```

```
  [
```

```
    ... Blue Cone Code
```

```
    Transform
```

```
    {
```

```
      rotation 0 0 1 .78
```

```
      children
```

```
      [
```

```
        ... White Cone Code
```

```
      ]
```

```
    }
```

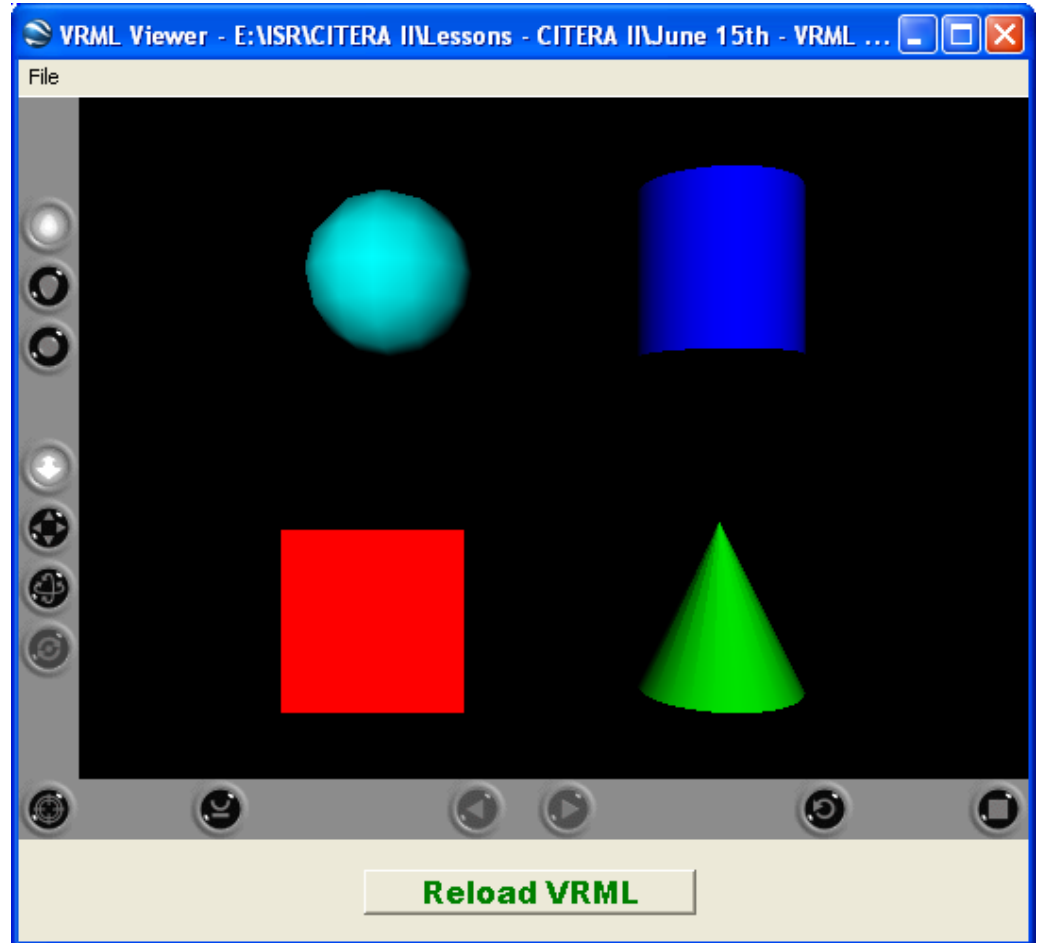
```
  ]
```

```
}
```

Program day3_9_Composite.wrl

Lessons: Basic Geometric Primitives

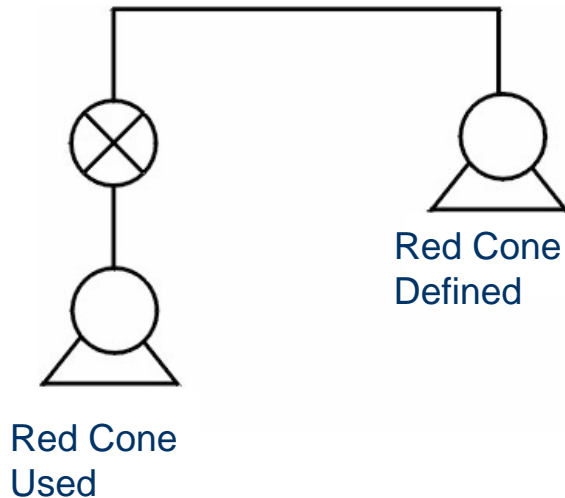
- Primitives
 - Cone
 - Box
 - Cylinder
 - Sphere
- Use to make more complex objects



Program day3_10_BasicPrimitives.wrl

Lessons: Using *DEF* / *USE*

- The ability to Reuse Portions of code
 - Use **DEF** to create a label for any VRML Node
 - Use **USE** to use that label later and save redundant code



```
#VRML V2.0 utf8

DEF RedCone Shape
{
  appearance DEF RedColor Appearance
  {
    material Material
    {
      diffuseColor 1 0 0
    }
  }
  geometry Cone {}
}
Transform
{
  translation 1 2 0
  children
  [
    USE RedCone
  ]
}
```

Program day3_11_DEF_USE.wrl

```
#VRML V2.0 utf8
```

```
Shape {
```

```
  geometry Sphere {}
```

```
  appearance Appearance {
```

```
    texture ImageTexture {  
      url "earth.jpg"
```

```
    }
```

```
  }
```

```
}
```

<http://www.die.net/earth/>



Program day3_12_Earth.wrl

- Use the **Background** node to texture map images to the background of your VRML scene
 - Your entire VRML scene fits inside of a BIG cube
 - You are texture mapping images to the 6 inside faces of the cube
 - VRML sets cube walls to the color black by default
- **Background** Node Default Fields
 - Most commonly adjusted fields
 - See Appendix A for full definition

```
Background
{
  backUrl    []
  bottomUrl  []
  frontUrl   []
  leftUrl    []
  rightUrl   []
  topUrl     []
}
```

Lessons: Using Background

1. Find image(s) for your 6 URLs
 - a. You can save them to your computer where your .wrl file is located.
 - b. Or, you can use the web url <http://webLocation/image.jpg>
2. Now use image(s) in the Background fields.
 - a. Place after the VRML header line

```
#VRML V2.0 utf8
Background
{
    backUrl    "stars.gif"
    bottomUrl  "stars.gif"
    frontUrl   "stars.gif"
    leftUrl    "stars.gif"
    rightUrl   "stars.gif"
    topUrl     "stars.gif"
}
... VRML Code
```

Program day3_13_BackgroundNode.wrl

“day3_12_Earth.wrl” includes
the code for our Earth model

```
#VRML V2.0 utf8

Shape {

  geometry Sphere {}

  appearance Appearance {

    texture ImageTexture {
      url "earth.jpg"
    }

  }

}
```

Lessons: Inline

Inline the Earth model with a
sun model in a new file

```
#VRML V2.0 utf8

Inline {
  url "day3_12_Earth.wrl"
}

Transform {
  translation 10 0 0
  children [
    Shape {
      geometry Sphere { radius 2 }
      appearance Appearance {
        material Material {
          diffuseColor 1 1 0
        }
      }
    }
  ]
}
```

Program day3_14_EarthAndSun.wrl

- Lighting has no effect on a Texture mapped shape unless there is a Material Node
 - This is because emissiveColor gets set to black -> 0 0 0

```
#VRML V2.0 utf8

Shape {

  geometry Sphere {}

  appearance Appearance {
    material Material {}
    texture ImageTexture {
      url "earth.jpg"
    }
  }
}
```

- Create a static model of an atom with electron rings.
- Create a static model of the solar system. Can you model this to scale?
- Add texture maps to your planets.