

# 1 More Blender Training

**Graphics Course—Blender Connections** Try out the features described below. Write a brief report of your experience (easy, troublesome, helpful, unsuccessful, etc.) with each.

## 1.1 Hot Keys

I have purposely omitted coverage of most “hot-keys” (one or more presses of keys to activate operations). If you like, you can learn and use the hot-keys by observing their names on the right of the menu selections. You can also view Book II of the Blender reference manual which is a comprehensive reference for all the Blender functions together with their hot-keys.

One easy-to-remember hot-key is the space bar pressed while the cursor is in the 3-D View window. It pops up an expanding tree of menus which you navigate by dragging the mouse and the left-clicking just once on the selection you want. If you like, use it in place of my menu instructions.

## 1.2 (Manipulators or “Widgets”)

Start Blender, after exiting your old Blender instance if necessary. It starts with a cube named Cube in the scene, Cube is selected, and Blender is in Object Mode. In Blender, the selected object is called **the active object**.

Blender also starts with a 3d View window (top 3/4) and a Buttons window (bottom 1/4).

Manipulators are a really cool feature that has been in commercial 3D graphics software for at least a couple of decades and is now in the new version of Blender.

1. First, (unrelated to Widgets) press and drag the Middle-Mouse-Button to first switch the 3D View from Top View to User View and then to make your view move around the world as you drag the mouse with the middle button down.
2. Second, notice, centered in the cube, the red, green and blue axes ending with with small red, green and blue squares. They comprise the **scale manipulator**. After you Left-Mouse-Click on one of the small squares, mouse motion will specify a scaling transformation (in the direction X, Y or Z corresponding to the red, green or blue color) and apply it to the object. Try it!
3. While the mouse controls the transformation parameter, the object is wire-framed in white. Let’s call the displayed transformation “tentative”.  
Click the Right-Mouse-Button to abandon the tentative transformation and quit the use of the manipulator you had activated. Try it! (Good user interface design always permits the user to back away from mistakes.)
4. Start another transformation and then click the Left-Mouse-Button to multiply the tentative transformation into the object’s transformation. The changes of the numbers in the Transformation Properties dialog will remain after you click the Left-Mouse-Button.
5. Third, you can enable any combination of scale, rotation and translation manipulators: (1) In the 3D View window’s menu bar, Shift-Left-Click on the red triangle, green circle and blue square icons; or (2) use the Control-Space hot-key then Left-Mouse-Click in a selection. (The hand icon disables and re-enables the entire manipulator feature.)
6. Try selecting them all (Combo). Use them them one at a time—The arrows do translations and the 3 colored semicircles do rotations. The inner white circle does translation in the plane you are viewing; the outer white circle does rotation around the viewing direction.
7. Finally, after you have rotated the cube, experiment with Local widget orientation instead of Global. Briefly describe the difference in your report.

8. **Individual mouse controlled transformation operations.** The older way to manipulate an object is to activate **Grab** from the 3D View window in any of these ways: (1) menu: (Object)/(Transform)/(Grab/Move). (2) Hot-key g. (3) Press drag and release the Left-Mouse-Button.

Here is another neat user interface feature: If you press and hold the Left-Mouse-Button, and quickly make a roughly circular motion, Blender will activate rotate transformation mode.

Subsequent mouse motion will rotate the object around your viewing direction. As before, Left-Click installs the transformation and Right-Click abandons it. (Consistency is good user interface design.)

9. **Undo Editing feature** In edit mode, the **Mesh** menu has an undo selection.

### 1.3 Names and New Object Creation

Exit and restart Blender..The default Cube is active in Object mode. In the Buttons Window observe the two **Panels: Link and Materials** and **Mesh**.

1. Observe in the Link and Materials panel the **Names** of (1) the original cube's **Mesh** data block "ME:Cube" and (to the right) the cube's **Object** data block "OB:Cube".
2. Please Left-Mouse-Click in the 3d View window with the mouse pointer about 5 units to the left (negative X direction) and 4 units back (positive Y direction). This moves the **3D Cursor** to a new position. Write in your report a description of the 3D Cursor including its colors. It is moved within the plane that the 3D View is currently facing orthogonally.
3. Create a new cube centered at the 3d Cursor by selecting (Add)/(Mesh)/(Cube) from the top menu bar. (Or press space and go from there..)

The new cube appears in the 3D View. The new cube is now **active** instead of the old cube. Observe that you can tell which object is active three ways: (1) the selected object is displayed more colorfully (in lavender), (2) the name Cube.001 is printed in the lower left corner of the 3D Display, and (3) in the Link and Materials Panel, names of the data blocks for the selected object are shown, "ME:Cube.001" and "OB:Cube.001".

4. (Object Mode versus Edit Mode) Observe now that after Blender created the new cube Cube.001, Blender automatically switched from **Object Mode** to **Edit Mode**. The **Mode** is displayed in the mode selection button with its icon (3 axes) and name (Edit Mode).

Furthermore, all 8 of the Cube.001's vertices are **selected**: You can tell because they are displayed in yellow.

5. In Blender, the **Buttons** window holds subwindows which interface to those *editing operations* that apply to the selected object. One of the attributes of a block (like a cube object) is its name.
6. Please switch the mode back to Object Mode for now, by pulling up the Mode menu and selecting Object Mode. The whole Cube.001 object is still selected but its vertices are not.

So, with Cube.001 selected, observe the "Link and Materials" submenu in the Buttons window. The edit menus like "Link and Materials" appear when the Editing Panels are activated. Please observe that in the menu of the Panels button, "Editing" is checked. The 6 square buttons to the right of the Panels button are shortcuts to the 6 kinds of panels.

7. ("A rose by any other name would smell just as sweet.") Observe again the names "ME:Cube.001" and "OB:Cube.001" in the Link and Materials panel in the Buttons window. They are *just names* stored as text within the mesh data block and object block data structures. (They are probably also stored in an index data structure so the software can efficiently locate a data block by name. I haven't checked by reading enough of the Blender source code to be sure!)

Please position the mouse cursor over each name and wait a second or two. Note (if you haven't already) that when you position the mouse cursor over a user interface element, it (1) changes color and (2) after a second or so, a yellow information "balloon" appears near it.

Blender automatically creates new names (like Cube.001, Cube.002, etc.) for new objects. You can change the name of the active object or data block by Left-Mouse-Click in the box holding the name and then editing

the text. Please *change* these names to the more useful ones “ME:BackCube” and “OB:BackCube” by a Left-Mouse-Click in each name, which enables editing (using backspace, delete, the arrow keys, and letter/number keys of course). Finish each edit with Enter.

When the *name* of an existing block is changed, the block itself (which you had seen in the Schematic and Outliner views of the Outliner window) **RETAINS ITS IDENTITY**.

## 1.4 Parenting and the Hierarchical Model

1. Now that you have two cubes and can modify the transformation attached to each object, make sure Blender is in Object mode and select (make active) the new cube named BackCube.
2. From the Object menu, activate the Transformation Properties dialog (if you haven’t already.)
3. In the Transformation Properties dialog, observe the Parent Object Box on the right near the top. It contains “Par:” followed by an empty space.

Check again that the Transformation Properties apply to “OB:BackCube”.

Make object BackCube become a child of the object named Cube by Left-Click inside the parent box to edit in the name “Cube”. This will make Cube be the (one, single) parent of BackCube. Press Enter when you are done typing “Cube”. (And of course, typing mistakes can be corrected with backspace, delete, arrow keys and retyping.)

Now you can observe the effect that BackCube is a child of Cube:

4. Select and then translate and rotate Cube (the parent). You should observe that BackCube (the child) is transformed along with its parent.
5. Select and then translate and rotate BackCube (the child). Observe that the transformation is not applied to any other object.
6. After you have transformed BackCube directly, reselect its parent Cube and apply some transformations. Observe that the transformations that you supplied directly have still been applied, and the new transformations you applied to the parent are “inherited” by the child.
7. Click on the Window Type button at the far left of the Buttons Window menu bar, and change the Buttons window to an Outliner Window.
8. Observe in the Schematic view that there is a link **from** the BackCube (child) object block **to** the Cube object block (parent). (The object blocks are those with the small 3-axis icons.) In Blender Schematic terminology, the Cube object block is **USED BY** the BackCube. That is, to render BackCube, the transformation from the parent is used after BackCube’s own transformation.

The end of the link going **from** a block is displayed with a small square colored with the color of the block that is being used. The end of a link going **to** a block is displayed by filled-in black half-circle.

9. Now, switch the view to Outliner and expand the outline by Left-Clicking on the right-pointing triangles. Expand and observe all the rows underneath the object block named Cube.

You should observe that there are two blocks directly under object block Cube: (1) the mesh named Cube and (2) the object named BackCube. Object BackCube has its own mesh, which is named BackCube.

## 1.5 Views and Rendering

1. (Different Views in the 3D View window.) The default **View** of the 3d View window is **Top** view. Observe in the lower left corner of the 3d View window two little colored coordinate axes colored red for X and green for Y. Top view means the 3d viewer is looking down from the positive Z axis in the direction of the negative Z axis. Just like the effect of the default viewing transformation (the identity matrix) in OpenGL! Unlike OpenGL, unfortunately, the view from the Z direction is called top, whereas in my OpenGL lectures, I called the view from the Z direction “front”.

The “User” view takes advantage of a nice technique for 3D realism: The user can move his/her view around the objects. Please move the mouse pointer into the 3D View, press and hold the Middle-Mouse-Button and drag the mouse around.

It should now be apparent that the two cubes’ centers are both in the XY-plane.

You can also see (1) the one camera and (2) the one lamp which are also within the Scene. The dashed line with the yellow circled dot at one end represents the lamp.

You can go back to the Top view by selecting it from the View menu.

2. (Perspective and Orthographic Views) Select the “Perspective” check item in the View menu.

Observe how the perspective projection changes as you move your view around by Middle-Button-Hold-Drag in the 3D View window.

Next, select the “Orthographic” check item, and observe the scene this time with orthographic projection. (Remember: Orthographic means parallel projection with the projection lines parallel to the projection plane. Mathematically, when the projection plane has equation  $Ax + By + Cz + D = 0$ , the direction of projection will be represented by the point at infinity with homogeneous coordinates  $(A, B, C, 0)$ .)

Please alternate between orthographic and perspective by selecting the respective check items (better: Num Pad 5 hotkey, make sure NUM-LOCK is enabled). Also combine that with top viewing, selected with the menu (or the Num Pad 7 key).

3. (ViewPort Shading) In the 3D View window menu bar, find the small menu button with the brown cube-like icon near the middle, just right of the mode menu button. Left-Hold on this button to see the the 5 “Draw Types”. Try them one by one.

Note the differences you can see between wire-frame and solid shading especially with perspective Top viewing.

4. (Camera View) Select (View)/(Camera) from the 3D View window menu bar.

5. Select the Camera object by RIGHT-Mouse-Click on the outermost border rectangle (solid) of the Camera view.

Observe that in the Buttons window, the Link and Materials panel now shows the camera data block “CA:Camera” and the camera object block “OB:Camera”.

Also, the second panel in the Buttons window is now a “Camera” panel.

6. You can use the “Camera” panel to change the *projection transformation* used by the camera:

- (1) Switch back and forth between the orthogonal and the default of perspective projection by Left-Mouse-Click on the “Ortho” button.

- (2) With perspective projection, vary the “Lens” parameter and observe the changes.

- (3) Make sure the ViewPort shading is set to solid (or shaded or textured). Use editing to set the number called “ClipSta” (for Clip Starting Distance) to 18.0 and press enter. Observe the camera view. Now repeatedly Left-Mouse-Click on right-pointing arrow next to the number to slowly increase it. Observe how more and more of the scene comes into view!

The “ClipSta” number is called the “Far clipping plane distance” in OpenGL.

7. Experiment with different camera settings with orthographic projection too.

8. In every 3D Viewer view type, including Camera, you can select and manipulate the objects. The mode must be Object mode in order to change which object is active. The active object can the be selected by a **Right**-mouse-click on it.

9. **Rendering** creates nice images that can be output to files.

First, redisplay the Buttons window. Select menu item (Panels)/(Scene)/(Render) (or use F10). Now press the big Render button.

Homework Question 1: It's a fact that 3D rotations in the X, Y, and Z directions do not commute (an X rotation followed by a Y rotation is different from the same Y rotation followed by the same X rotation.) Yet, Blender's (Object)/(Transform Properties) dialog box for an object (like a cube) in Object mode displays 3 separate edit boxes for X, Y, and Z rotation.

Determine by experimentation and observation: When X, Y, and Z rotations are specified, which rotation is done first, second and third? Does it matter in which order they are done? Explain briefly how you figured it out, including what amount of rotations you tried.

## 2 Homework on Parametric Representation and Clipping

- Derive and write formulas for  $a$ ,  $b$ ,  $c$  and  $d$  in the problem:  
Suppose a line segment is given by its endpoints  $(x_0, y_0)$  and  $(x_1, y_1)$ . What are the coefficients  $a$ ,  $b$ ,  $c$  and  $d$  so the parametric representation is  $P(u) = (a + bu, c + du)$ , directed so  $P(0) = (x_0, y_0)$  and  $P(1) = (x_1, y_1)$ ?
- This problem is about the line determined by points  $(x_0, y_0)$ ,  $(x_1, y_1)$  of problem 1 and its intersections with horizontal and vertical lines.
  - Write the condition to test to tell if the intersection with the horizontal line with equation  $y = y_W$  exists and is unique.
  - When this condition is true, what is the value of parameter  $u$  for the intersection?
  - What are the  $x, y$  coordinates of the intersection?
  - Repeat the above 3 parts for the vertical line  $x = x_W$ .
- Derive a formula for the intersection of the line given by  $y = mx + b$  and the line given (in parametric form) by  $P(u) = P_0 + u\Delta P$  where  $P_0 = (x_0, y_0)$ ,  $P_1 = (x_1, y_1)$  and  $\Delta P = P_1 - P_0 = (\Delta x, \Delta y)$ .
- Suppose two lines are given by parametric representations  $P(s) = (a + bs, c + ds)$  and  $Q(t) = (e + ft, g + ht)$  ( $a, b, c, d, e, f, g$  and  $h$  are constants.)
  - Write the two simultaneous equations to solve to find the (two) parameter values  $s'$  and  $t'$  so the intersection is  $P(s') = Q(t')$ .
  - Solve for  $s'$  and  $t'$  in terms of the given data (consult a math review book on simultaneous linear equation solving if necessary).

### Notes on clipping, parametric form of line, Liang-Barsky method

The clipping problem: what it is

Clipping window and line segments

Finding intersection of line with a horiz. or vertical window boundary

"High school method"

use  $y=mx+b$  or  $x=(1/m)y+a$ , with known  
x value or y value for window edge

Cohen-Sutherland simplification

classify the 9 regions defined by the 4 window edge lines  
with a 4 bit code. If the endpoints of a line fall in  
regions with the bits in one of the positions both equal  
to 1, then the line is outside the window.

Parametric form method (important for many reasons)

Parametric expression for line

Let (2 dim vector)  $\Delta = P_2 - P_1$

$P(u) = P_1 + u \Delta$   
 $= P_1 + u (P_2 - P_1)$   
 $= (1-u)P_1 + uP_2$

note that  $\alpha_1=(1-u)$  and  $\alpha_2=u$  add up to 1.

Significance of parameter value  $u$

Liang-Barsky clipping

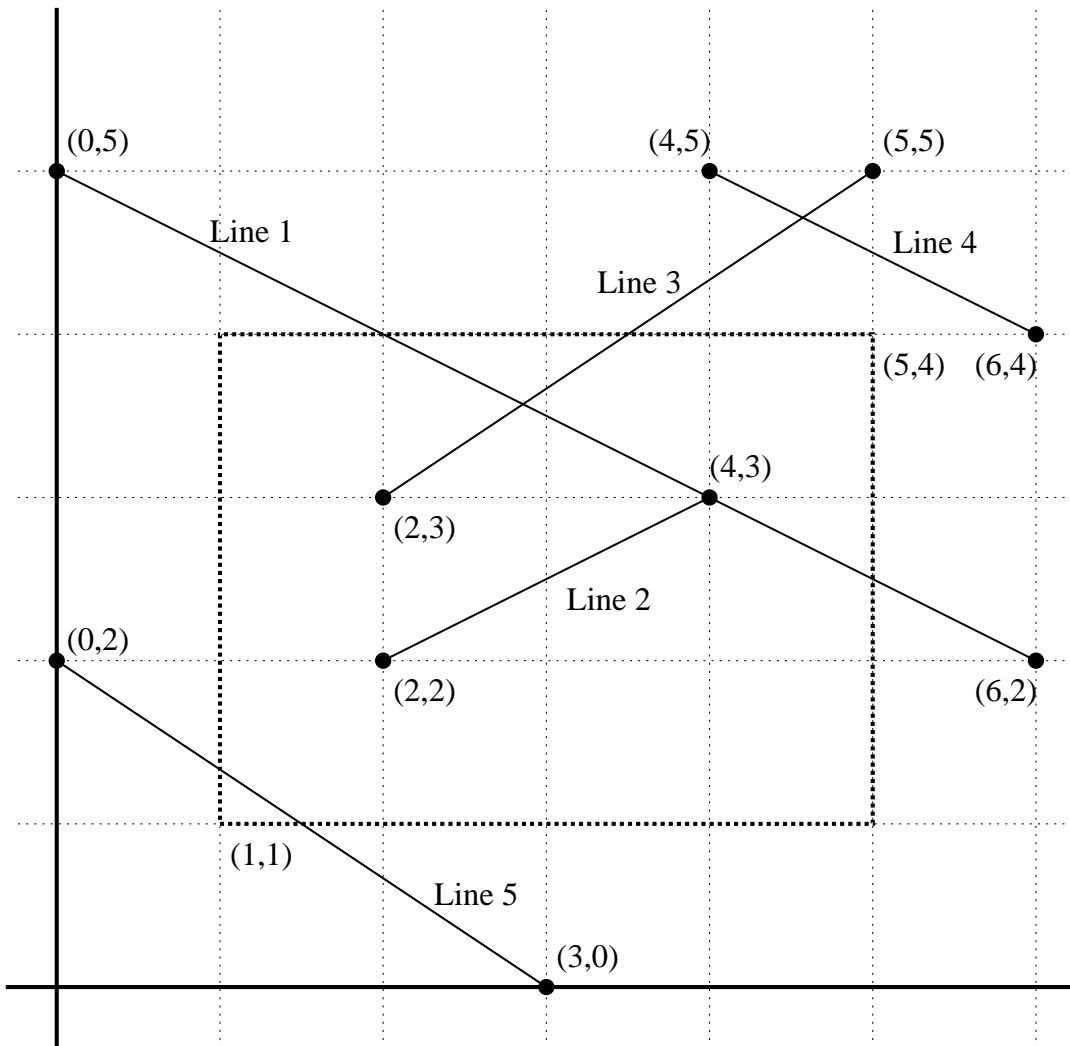
Find largest interval  $[u_1, u_2]$  of  $u$  for which the  
corresponding points in the line segment are in the window  
These  $u_1$  and  $u_2$  values are computed by solving 6 linear inequalities  
The first two are  $0 \leq u$  (applies to  $u_1$ )  
and  $u \leq 1$  (applies to  $u_2$ )

Algorithm outline

```
Initialize  $u_1:=0$ ;  $u_2:=1$ ;  
For each of 4 inequalities  
   $x_{\min} \leq x_1 + u \Delta x$   
   $x_1 + u \Delta x \leq x_{\max}$   
   $y_{\min} \leq y_1 + u \Delta y$   
   $y_1 + u \Delta y \leq y_{\max}$   
{  
  if the inequality is equivalent to  
     $u \leq \text{Something}$   
    then  $u_2:=\min(u_2, \text{Something})$ ;  
  else if the inequality is equivalent to  
     $u \geq \text{Something}$   
    then  $u_1:=\max(u_1, \text{Something})$ ;  
  if  $u$  has changed and  $u_1 > u_2$  then stop;  
}  
draw line from  $P_1 + u_1 \Delta$   
to  $P_1 + u_2 \Delta$ 
```

Refer to the figure below.

5. For each of the five lines do the following:
  - (a) Write the parametric form for each of the 5 lines. Check that the correct endpoints are obtained when the parameter is 0 and is 1.
  - (b) Write the minimum and maximum,  $x$ -coordinate and  $y$ -coordinates for the dotted rectangle to be used as a clipping window.
  - (c) Write the four inequalities that are in the derivation of the Liang-Barsky clipping algorithm, which limit the parameter to give points within the clipping region specified by each of four rectangle sides. Identify which of the inequalities are equivalent to " $u \leq \text{Something}$ " and which are equivalent to " $u \geq \text{Something}$ ".
  - (d) Starting with  $u_1$  initialized to 0 and  $u_2$  initialized to 1, show how the processing of each inequality updates or leaves unchanged either  $u_1$  or  $u_2$ , and find the final values for  $u_1$  and  $u_2$  after all the processing.
  - (e) Show how the final values of  $u_1$  and  $u_2$  indicate whether the clipped line would appear in the clipping window at all. For lines that do appear, calculate the endpoints of the line segment after clipping (evaluate the parametric form on  $u_1$  and on  $u_2$ ) and confirm that your result is consistent with the diagram.
6. Use the parametric method to compute the intersection of line 1 with *each* of the three lines line 2, line 3 and line 4. (For each, solve one pair of simultaneous linear equations.) Observe that something goes wrong for the intersection of line 1 with line 4. Write why it went wrong. For the others, confirm that your calculations are consistent with the diagram and if they are not, try to fix them and ask for help from me.



**Practice problems (do not hand in):**

1P Write formulas for the entries of the inverse  $A^{-1}$  of the  $2 \times 2$  matrix

$$A = \begin{pmatrix} a & c \\ b & d \end{pmatrix}$$

and then show *two separate* calculations that demonstrate that

$$AA^{-1} = I \text{ (the identity matrix)} \quad \text{and} \quad A^{-1}A = I.$$

2P Let

$$A = \begin{pmatrix} a & c \\ b & d \end{pmatrix} \quad \text{and} \quad P = \begin{pmatrix} p & r \\ q & s \end{pmatrix}$$

- What matrix represents the linear transformation that maps  $(1, 0)^t \rightarrow (a, b)^t$  and  $(0, 1)^t \rightarrow (c, d)^t$ ? (The symbol “ $\rightarrow$ ” is read as “to.”)
- What matrix represents the linear transformation that maps  $(a, b)^t \rightarrow (1, 0)^t$  and  $(c, d)^t \rightarrow (0, 1)^t$ ? Write the calculation to check that it really does this.
- What matrix represents the linear transformation that maps  $(a, b)^t \rightarrow (p, q)^t$  and  $(c, d)^t \rightarrow (r, s)^t$ ? (Big hint: matrix equation  $NA = P$  has the solution  $N = PA^{-1}$ .) Calculate  $A^{-1}$  and the matrix product  $N$ . Write the calculation to check that it really does the given mapping.

- (d) What matrix represents the linear transformation that maps  $(p, q)^t \rightarrow (a, b)^t$  and  $(r, s)^t \rightarrow (c, d)^t$ ? Write the calculation to check that it really does the given mapping.

### 3 Previous Blender training, for reference and with some errors corrected

1. When Blender starts, the default scene contains a camera, a lamp and a cube. The cube is **selected**. The **MODE** is **object mode**. Object mode means that the entities that you can edit apply to the *whole* selected object. Select (from Object menu) (Object)/(Transform Properties) to view the **transformation** attached to the cube. Edit this transformation to move, rotate and scale the whole cube.
2. The big window showing the cube is a **3d View** window. The bottom window is a **Buttons** window. Use the **Window Type Selection Button**, far left button on the top of the Buttons window, to view the 15 different types of windows Blender has. Select the **Outliner** (5th from the top) to replace the Buttons with an outliner window.

Blender (like all serious OpenGL based software) implements its own **hierarchical modelling data structures** to represent scenes and objects. Blender's **outliner** displays the major blocks of this data structure and the links. Of course, links are implemented by **pointers**.

You can **Pan around** in the Outliner window using Shift-MiddleMouseHold-Drag. Try it. (With the mouse cursor in the Outliner window, press and hold the shift key with your left hand. Then press IN and hold the wheel or middle mouse button depending on the species of your mouse. Then move the mouse.)

Observe all the boxes and their names.

You can also zoom in and out for viewing the Outliner window by Control-MiddleHold-Drag. Zoom in enough to see the names and icons in each outliner box.

Look for the two different boxes named "Cube".

The one with the 3-coordinate-axes-arrows icon is the **Object block** for the cube named Cube.

The one with the triangle icon is the **MEsh block** for the cube named Cube.

3. In Blender, the Right-Mouse-Button is used for **Selecting** things.  
Please select the **Mesh block** for the cube. Notice that the name of the selected block, "MECube" appears just after all the buttons on top of the Outliner window, towards the right.  
Now, possibly after panning and/or scaling, select the **Object block** for the cube. Notice its name, "OBCube", has replaced the name of the previously selected block "MECube".
4. What you examined in the outliner window is the **Oops Schematic view** of some data. Database people today talk about different **views** of portions of the same data. The outliner window implements an alternative view called the **Outliner**.  
Select the (View)/(Outliner) menu button to switch to the **Outliner** view.
5. The Outliner view works like the Windows Explorer viewer of folders and files, and like the "Dom" tree viewer of html documents in some web browsers.

Initially, observe only one line: A right-pointing triangle and the name "Scene". This represent the **root** of the Scene data structure.

Please **TYPO:Left-Click** on that right-pointing triangle. Observe (1) the triangle changes to a down-pointing triangle, and (2) the **children** of the Scene node are now displayed.

Please pan up and down by doing Middle-Mouse-Hold-and-Drag with the cursor in the left scroll bar.

Whenever you see a right-pointing triangle, **TYPO:Left-click** on it to observe more details in the tree.

6. Make sure you can observe the Cube **object block** (the three axis arrows icon followed by “Cube”) and the Cube **mesh block** (the triangle icon followed by “Cube”) at the same time.

Please Left-Mouse-Click on the Cube mesh block icon (the triangle) again and again. Observe that the **Object Mode** and **Edit Mode** of the 3-d view window switch back and forth.

7. The lesson to be learned is that when Blender is in **Edit Mode** of a cube object, editing operations apply to the vertices, edges, faces, individually or in selected groups. In this mode, Blender lets you edit the details within the Mesh data structure.

You will move a vertex in the mesh by **editing the TRANSFORMATION** that is associated to that vertex.

On the other hand, when Blender is in **Object Mode** of a cube object, editing operations apply to the **WHOLE** cube object. You can edit the transformation applied to the whole cube to move it around.

You can also edit the **COLOR** and **MATERIAL** attached to the whole cube.

More will follow in new assignments!