**Project Reflection: Design Decisions**

For this project, there were many factors to consider before creating my three-dimensional scene. First, I needed to assess my experience with the OpenGL API—of which I had none. To address this, I decided to keep the objects in my scene simple so I could focus on understanding how the API functions and putting it to use effectively. I chose simple geometric objects located near my desk and arranged them in close proximity to one another. Each object had unique characteristics that influenced my choices.

The block man was selected for its complexity, which helped me understand the x, y, and z positions in rendering the scene. Each component of the block man was given its own scale, rotation, and position to replicate the picture I submitted in Module One. I also wanted to add a playful element to the project, and creating a block man figure provided just enough challenge while remaining feasible within the allotted time. Its plastic material gives it a shiny appearance when hit by light. However, I realized that most of my chosen objects were plastic, so I made the block man’s clothing a cloth material. This allowed me to experiment more with shaders, materials, and lighting effects.

My second object was a simple aluminum can, which was much easier to build. While I couldn’t find a texture that matched the one in my reference image, I found a suitable substitute that made for an interesting choice. The can’s side, is shiny but not overly reflective, while the top of the can is both shiny and reflective. To replicate this, I used two different types of cylinders with distinct materials for each section. This effectively recreated the light interactions. As you move the camera around the scene, you can see how the top of the can catches light from both primary light sources, depending on the perspective.

The third object was a pair of headphones that I keep near my desk. Like the block man, the headphones are made of plastic, though their finish is more subdued. To replicate this, I created a material called “matte” to mimic the dull finish found in real-world objects of the same type. This material reflects light but lacks shininess or specular effects. Unfortunately, I ran out of time to flesh out this object as much as I had hoped. Additionally, I encountered distortion issues when attempting to elongate and flatten a half torus. While scaling the object worked to some degree, at certain angles, the proportions appear off. I explored creating custom shapes but found that calculating the vertices and building a custom mesh would take too long. This is something I hope to revisit and improve upon in the future.

The final object in my scene was the lamp base that sits on my desk. This object, also made of plastic, has a darker finish compared to the headphones. I chose to render only the base of the lamp to prevent the top from interfering with the rest of the scene. The lamp base is composed of several basic geometric shapes, and its dark blue plastic material creates an appealing effect when struck by light. I used a plastic material for this object and positioned a light source directly above the base to mimic the effect of the lamp’s light bouncing off its surface.

To navigate the scene, users can move the mouse to adjust the camera perspective and switch between orthogonal views by pressing the 1, 2, or 3 keys. The Q and E keys move the camera along the Y-axis, the A and D keys move along the X-axis, and the W and S keys adjust movement along the Z-axis. Additionally, users can adjust the camera’s movement speed by scrolling the mouse wheel up or down.

To keep my code modular and easy to understand, I created custom functions to encapsulate the information for each object I built. These functions make the code cleaner and more reusable. For instance, if I needed multiple soda cans, I could create a variable to hold the soda can method and reuse it in different scenes. To enhance modularity further, I could move these functions from the .cpp file into a .header file, allowing them to be imported into other applications or locations within the same project.