OpenGL II: Basics
Vertex Array Object

- Bundles all vertex data (positions, colors, ..,)
- Get name for buffer then bind

```c
gluint abuffer;
glGenVertexArrays(1, &abuffer);
glBindVertexArray(abuffer);
```

- At this point we have a current vertex array but no contents
- **Use of** `glBindVertexArray` **lets us switch between VAOs**
Buffer Object

- Buffers objects allow us to transfer large amounts of data to the GPU
- Need to create, bind and identify data

```c
gluint buffer;
glGenBuffers(1, &buffer);
glBindBuffer(GL_ARRAY_BUFFER, buffer);
glBufferData(GL_ARRAY_BUFFER, sizeof(points), points, GL_STATIC_DRAW);
```

- Data in current vertex array is sent to GPU
Initialization

- Vertex array objects and buffer objects can be set up on `init()`
- Also set clear color and other OpenGL parameters
- Also set up shaders as part of initialization
  - Read
  - Compile
  - Link
- First let’s consider a few other issues
Coordinate Systems

- The units in points are determined by the application and are called object, world, model or problem coordinates.
- Viewing specifications usually are also in object coordinates.
- Eventually pixels will be produced in window coordinates.
- OpenGL also uses some internal representations that usually are not visible to the application but are important in the shaders.
OpenGL Camera

- OpenGL places a camera at the origin in object space pointing in the negative z direction
- The default viewing volume is a box centered at the origin with sides of length 2
Orthographic Viewing

In the default orthographic view, points are projected forward along the $z$ axis onto the plane $z=0$. 
Viewport

- Do not have use the entire window for the image: `glViewport(x, y, w, h)`
- Values in pixels (window coordinates)
Transformations and Viewing

- In OpenGL, projection is carried out by a projection matrix (transformation)
- Transformation functions are also used for changes in coordinate systems
- Pre 3.0 OpenGL had a set of transformation functions which have been deprecated
- Three choices
  - Application code
  - GLSL functions
  - vec.h and mat.h