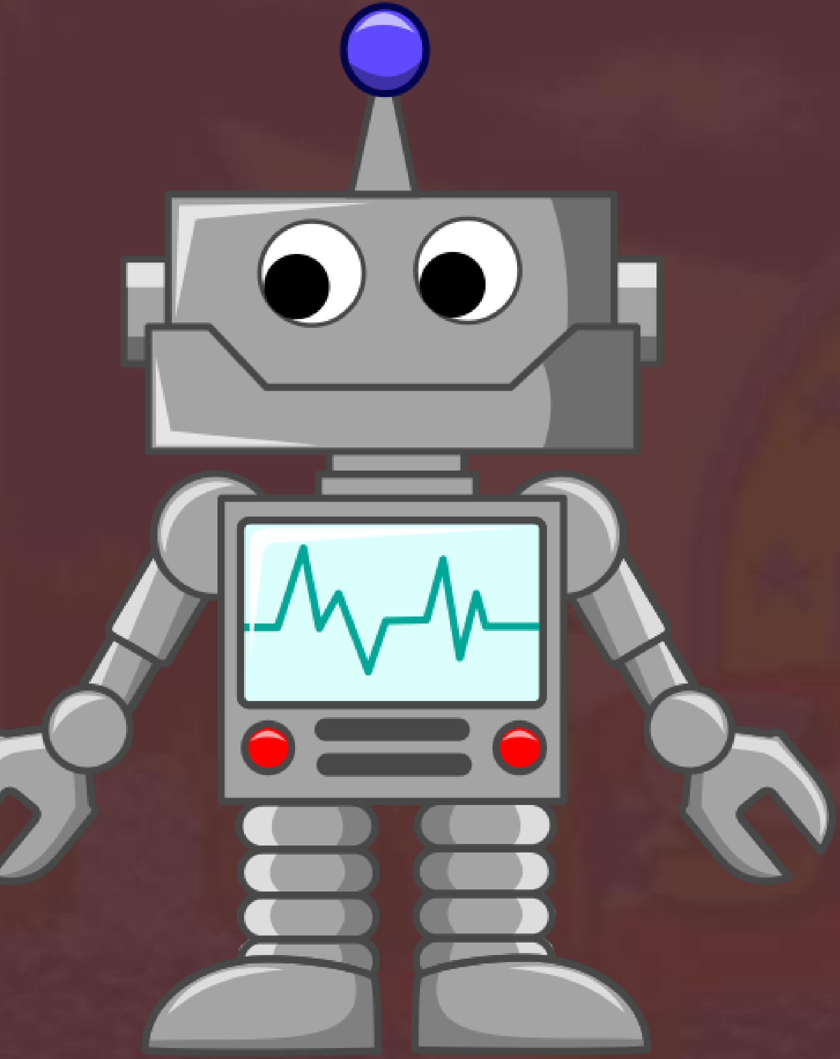
State Machines & Drawing in OpenGL

Game 300

Objectives

- What is a state machine
- Set the basic states to an OpenGL application.
- Understand the way the GPU processes draw commands

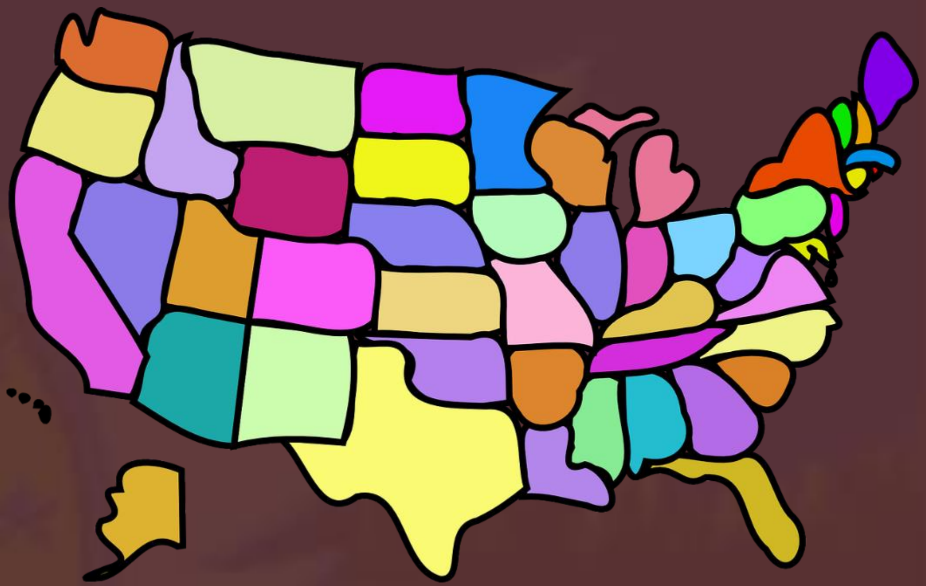
State machine



- A state machine is a program which keeps track of its current state and completes some sort of functionality based on that state.
- Holds variables with pre-determined acceptable values.
    - Based on the values assigned it alters the way it functions.
- You will learn more about state machines inside of your AI class with finite state machines.

States

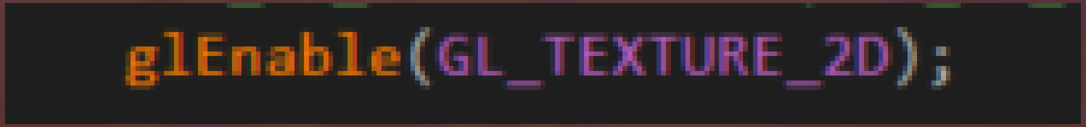
- OpenGL is a state based API or a state machine.
  - It keeps track of a lot of data hidden within it like:
    - Colour to draw with
    - Does the program allow depth
    - How far does the depth get processed
    - Etc... There is a lot going on.



- OpenGL allows programmers to change the settings of how it's API is going to function.
  - This gives us some control over how the API operates and processes our future commands.

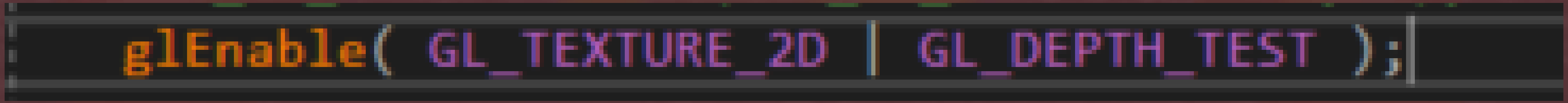
Binary States

- The most basic state setting we can change are the states which are Binary.
  - They are on or off, 1 or 0.
- To set or check these settings we can use the glEnable and glDisable functions.
  - glEnable(Glenum state);
  - glDisable(Glenum state);
  - bool glIsEnabled(Glenum state);



```
glEnable(GL_TEXTURE_2D);
```

- GLenum is the state that we want to enable or disable.
  - They are all values part of a large series of defines
  - Glenum is in fact not truly an enum… it is actually an unsigned int.
    - This allows you to set multiple states by using our bitwise OR |.



```
glEnable( GL_TEXTURE_2D | GL_DEPTH_TEST );
```

Getting Variable States

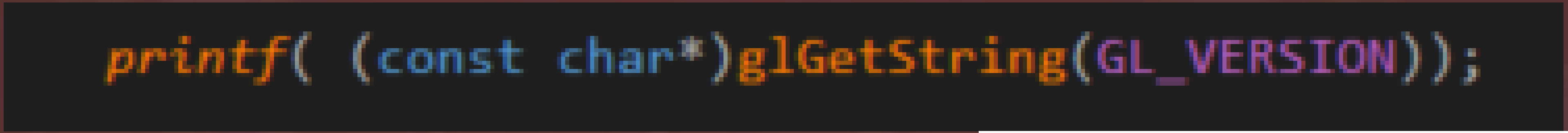
- Not all states are binary. (on or off)
  - If you needed to retrieve the state of a current setting of OpenGL you can use the series of glGet functions:
    - glGetIntegerv(GLenum name, GLint * params);
    - glGetFloatv(GLenum name, GLfloatt * params);
    - glGetBooleanv(GLenum name, GLboolean * params);

- Analyzing the above supplied API functions we see that the function requires a name as a parameter as well as a pointer to the variable type.
  - The name value is the state you want to query.
  - The params value is a local variable you create which will be populated with the current state that OpenGL has presently set.

String States

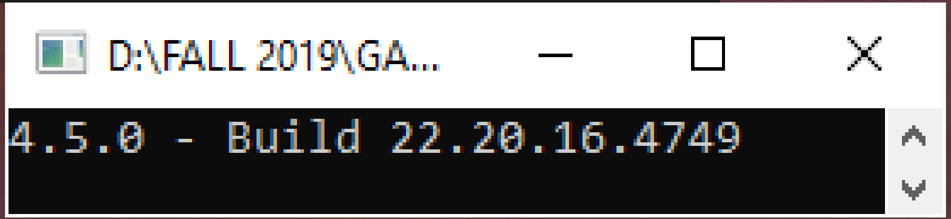
- We can retrieve a string state using the following:
  - const Glubyte* glGetString(Glenum name);

- * Note that strings are different as they return a reference to the value instead of having to pass one in as a parameter.
  - We can use this to determine things like the Graphics card used, or the max supported version of openGL:



```
printf( (const char*)glGetString(GL_VERSION));
```

  - This can be helpful in determining whether the users machine is capable of running your game.



```
4.5.0 - Build 22.20.16.4749
```

Setting States

- Some of OpenGL states are not able to be set.
  - Take for instance the previous state which returns the vendor or current capabilities of the graphics card.
  - To avoid having programmers accidentally attempt to set states that they should not have access to, no specific setState function exists.
  - Instead many states have individual functions to set their values.

- Binary values however can be set using the glEnable() function.

States

- Some states you will be introduced to in this course are:
  - **GL_DEPTH_TEST**
  - **GL_TEXTURE_2D**
  - GL_FRONT_FACE
  - **GL_POINT_SIZE**
  - GL_ALPHA_TEST
  - **GL_VENDOR**
  - **GL_VERSION**
  - GL_VERTEX_ARRAY_SIZE

SUMMARY

- Learned to:
  - What a state machine is and why it’s important.
  - Set the basic states to an OpenGL application.

- For further support read chapter 3 up to page 50.