Introduction

Motivation

- Introduction to topics for this class
- Get the requisite software packages working

GPU

- GPU = Graphics Processing Unit
  - *Graphics* is a bit of a misnomer
  - Very powerful computational unit
- Most of this class $\rightarrow$ Using the GPU

Libraries

- How to interface with GPU?
- Several choices...
  - OpenGL
  - DirectX 11
  - DirectX 12
  - Vulkan
  - Metal (Apple only)

Libraries

- OpenGL / DirectX: Pretty similar (concept-wise)
  - Syntax is quite different
- DirectX12 / Vulkan/Metal: Also more-or-less equivalent

OpenGL

- Desktop: Windows, Mac, Linux
- Mobile: Android, iOS: OpenGL ES
- Web browser: IE, Chrome, Firefox, Opera, Safari (WebGL)
- Programming languages: C, Python, Java, Javascript, ...

DirectX

- Desktop: Windows
- Mobile: Windows Phone
- Xbox
- Programming languages: C++, C# (“unofficial” binding)

DX12/Vulkan/Metal

- Very low-level APIs
- Somewhat difficult to work with
- Designed for high efficiency, no bottlenecks
- Programming Languages: C/C++

This Class

- We'll use OpenGL + Python
- This will require a few tools...
- Create a folder for project files (we'll call this the "project folder")

SDL

- Download SDL2 Runtime Binaries: http://www.libsdl.org/download-2.0.php
  - Make sure you get 32 or 64 bit correct, depending on which Python you have
- Unzip the zip file
  - The only file we need is SDL.dll
  - Put it somewhere on your disk

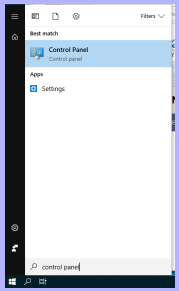
PySDL

- SDL is written in C
- We need a *binding* to Python
- Get pysdl2.zip from the class website
- Unzip it, put the sdl2 folder in your project folder
  - Careful! Not the pysdl folder itself – the sdl2 folder *inside* the pysdl folder!

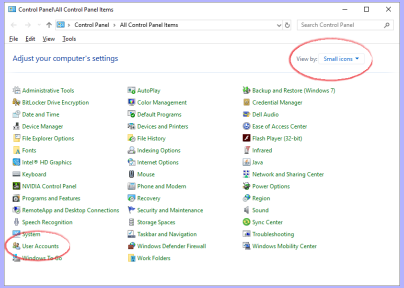
Environment Variable

- Set the PYSDL2_DLL_PATH environment variable to point to the folder where you put SDL.dll
- On the following slides...

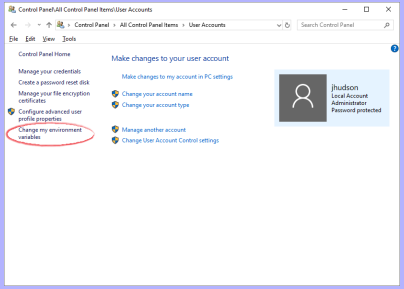
Setting Path



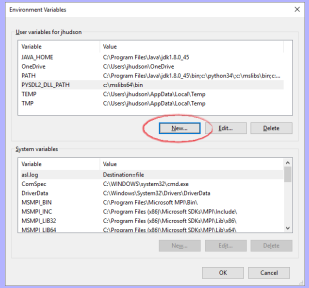
Setting Path



Setting Path

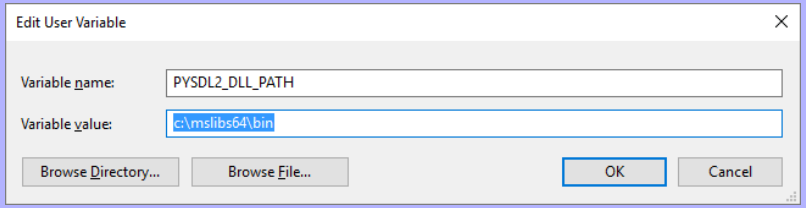


Setting Path



Setting Path

- Note: Don't put the SDL.dll name at the end of the path (we just want the folder name)



PyGL

- OpenGL is designed for C
- We need bindings to Python
- Obtain gl.py and glconstants.py from the class website
- Put these files in your project folder

Sound

- Perhaps we'll decide to include audio at some point
- Visit `http://www.libsdl.org/projects/SDL_mixer/`
- Download the runtime binaries (32 or 64 bit, depending on your Python version)
- Extract the .dll's from the zip file, put it in the same folder as SDL2.dll

Folders

- All of these should be in the same folder:
  - glconstants.py
  - gl.py
  - sdl2 (folder from pysdl.zip)
  - main.py (to be written)
  - assets (folder: We'll use this later for art assets)
  - shaders (folder: We'll use this later)

Ready to Go

- Now we're ready to go
- We'll first write a program that just puts a window on the screen
- See globals.py and main.py

Assignment

- If you have a laptop with hybrid graphics, *switch it to use the discrete GPU*
  - This is important!
- Submit the output (contents of “caps.txt”) of the test program (main.py) on the computer you will primarily use for this class
  - This is due *before the start of the next class session!!!*
  - Really easy!
  - This is to make sure you’ve gotten the packages set up!
  - And so I can make sure we don’t use capabilities that aren’t implemented on everyone’s GPU’s

Created using LaTeX.

Main font: Gentium Book Basic, by Victor Gaultney. See http://software.sil.org/gentium/
Monospace font: Source Code Pro, by Paul D. Hunt. See https://fonts.google.com/specimen/Source+Code+Pro and http://sourceforge.net/adobe
Icons by Ulisse Perusin, Steven Garrity, Lapo Calamandrei, Ryan Collier, Rodney Dawes, Andreas Nilsson, Tuomas Kuosmanen, Garrett LeSage, and Jakub Steiner. See http://tango-project.org