use Adobe Reader to complete

(Type in fields)

PA3 – FBX Converter

Student Information

Integrity Policy: All university integrity and class syllabus policies have been followed. I have neither given, nor received, nor have I tolerated others' use of unauthorized aid.

I understand and followed these policies: ● Yes ○ No

Name: Jacob Zackaria

Date: 10/24/20

Submission Details

Final ***Changelist*** number: 74441

Verified build: ● Yes ○ No

YouTube Link: https://youtu.be/sLX3b0hbepA

Required Configurations: Any - just need one that works

Discussion (What did you learn):

Runs in Debug (x86).
Learned about FBX Converter. Converting different models with different vertex, polygons and texture data. Learned th usage of FBX SDK.

Verify Builds

- Follow the Piazza procedure on submission
    - Verify your submission compiles and works at the changelist number.
- Verify that only MINIMUM files are submitted
    - No – Generated files
        - *.pdb, *.suo, *.sdf, *.user, *.obj, *.exe, *.log, *.pdb, *.db, *.user
        - Anything that is generated by the compiler should not be included
    - No – Generated directories
        - /Debug, /Release, /Log, /ipch, /.vs
- Typical files project files that are required
    - *.sln, *.cpp, *.h
    - *.vcxproj, *.vcxproj.filters, *.vcxproj.user, CleanMe.bat

Standard Rules

Submit multiple times to Perforce

- Submit your work as you go to perforce several times (at least 5)
    - As soon as you get something working, submit to perforce
    - Have reasonable check-in comments
        - Points will be deducted if minimum is not reached

Write all programs in cross-platform C++

- Optimize for execution speed and robustness
- Working code doesn’t mean full credit

Submission Report

- Fill out the submission Report
    - No report, no grade

Code and project needs to compile and run

- Make sure that your program compiles and runs
    - Warning level ALL …
    - NO Warnings or ERRORS
        - Your code should be squeaky clean.
    - Code needs to work “as-is”.
        - No modifications to files or deleting files necessary to compile or run.
    - All your code must compile from perforce with no modifications.
        - Otherwise it's a 0, no exceptions

Project needs to run to completion

- If it crashes for any reason…
    - It will not be graded and you get a 0

No Containers

- NO STL allowed {Vector, Lists, Sets, etc...}
  - No automatic containers or arrays
  - You need to do this the old fashion way - ***YOU EARNED IT***

Leave Project Settings

- Do NOT change the project or warning level
  - Any changing of level or suppression of warnings is an integrity issue

Simple C++

- No modern C++
  - No Lambdas, Autos, templates, etc…
  - No Boost
- NO Streams
  - Used fopen, fread, fwrite…
- No code in MACROS
  - Code needs to be in cpp files to see and debug it easy
- ***Exception:***
  - implicit problem needs templates

Leaking Memory

- If the program leaks memory
  - There is a deduction of 20% of grade
- If a class creates an object using new/malloc
  - It is responsible for its deletion
- Any ***MEMORY*** dynamically allocated that isn't freed up is ***LEAKING***
  - Leaking is ***HORRIBLE***, so you lose points

No Debug code or files disabled

- Make sure the program is returned to the original state
  - If you added debug code, please return to original state
- If you disabled file, you need to re-enable the files
  - All files must be active to get credit.
  - Better to lose points for unit tests than to disable and lose all points

UnitTestConfiguration file (if provided) needs to be set by user

- Grading will be on the UnitTestConfiguration settings
  - Please explicitly set which tests you want graded… no regrading if set incorrectly

Due Date

- Due during by milestone 1 (approx duration ~1 week)
  - Submit all files and directories to Perforce
  - Grading On MS1
- Submit all files and directories to Perforce
  - Create a directory called: ***Converte***r or ***PA3*** in your student directory
  - /student/<yourname>/PA3/…

Goals

- Write a standalone model converter
  - Takes FBX model and exports to ***YOUR*** custom format
  - Converts data to your model format
    - VBO runtime format
    - Any necessary headers for runtime
    - Adds the texture (data embedded into archive)
    - Converted into 1 archive (dependent on PA1)
- Export at least ***5 models*** with large polygon count
  - Required
    - Space_Frigate
  - One from each of the categories
    - Group A
    - Group B
    - Group C
    - Group D - Student's favorite models
  - Group D - models
    - We will have a Group_D directory in /Common
      - Students can push any model they would like to use
    - Or use anyone that meets the requirement
      - Find your models anywhere... make them or get them from the internet
      - Do not overload Group_D with too many models
      - Be selective - only models that you would convert
      - Requirements for Group D
      - At Greater than 1K Verts
    - Needs at least one Texture
      - Can have more one Texture (add a little complexity but it's worth it)
      - Needs vertex normals
      - FBX format
      - Modify existing game engine to load model from binary format file

Assignments

1. **FBX model converter:**
    - Create a ***standalone FBX converter*** to convert to your custom run-time file format.
        - Standalone executable
            - Create and check in the executable for the converter
            - Provide instructions how to convert the data from a command line
                - Yes - it needs to be professional and command line based
                - This way it can be added to a batch or script file
            - All models should be tessellated into triangles
        - Suggested formats - (***<u>You can do it anyway</u>***, as long as it's documented)
            - Parameter formats:
                - ***FBXConverter*** *< options >* ***inputModel.fbx outputfile.ext***
            - Additional parameters (not required) can be added to the converter suggestions:
                - ***–exportTexture***
                    1. Export texture separately
                - ***–includeTexture***
                    1. Include the texture in the output file
        - Supply a batch file or script showing your command line process
            - Script is expected to be working
            - Into your final binary package for the game
            - What is a batch file?
                - Command line *.bat file
                - Like the CleanMe.bat

2. **Export at least 5 models with large polygon count**
    - Idea to make this data driven without hand tweaking or modification
        - ***1 Required Models:***
            - ***Space Frigate*** – 298 polygons, textured, 777 vertices
        - One from each of the categories
            - Group A
            - Group B
            - Group C
            - Group D - Student's favorite models
        - Group D - models
            - We will have a Group_D directory in /Common
                - Students can push any model they would like to use

        - Or use anyone that meets the requirement
            - Find your models anywhere... make them or get them from the internet
            - Do not overload Group_D with too many models
            - Be selective - only models that you would convert
            - Requirements for Group D
            - At Greater than 1K Verts
        - Needs at least one Texture
            - Can have more Textures
            - Needs vertex normals
            - FBX format
            - Modify existing game engine to load model from binary format file
- Models need to have a texture
    - Simple model format
        - Single texture, vertex normal, vertex colors
            - Expected vertex data:
            - Vertex data – vx,vy,vz
            - Vertex normals – nx, ny, nz
            - Texture coordinates – u, v
            - Vertex colors – r,g,b,a (optional)

3. **Game engine needs to be modified to read your custom binary file**
    - Engine should be able to load file data once
        - Creating temp buffers for file
        - Load the data (VAO, VBOs, & Textures) into graphics memory
            - Remember you need to embed your texture into your archived data set
        - Free temp buffer
    - Be able to transform and move the model with a single transform
    - Setup rendering shader to display model with lighting
        - Using the OpenGL super bible code is allowed.

4. Write up document, video, and blogs

- ***Write up a quick description***
    - 2-x pages
        - How your converter works
        - Your format of the runtime
        - Description of the work you done
- ***Record the demo* (due with PA4 Viewer)**
    - Video demo of key aspects of your code (code review) and a command line conversion of data
        - Make sure each model is visible in your engine
            - Either drive the camera or spin the models
                - Show the complete model with textures
        - Please make a batch program to convert the models
            - Repeated process to successfully convert the fbx model into your format
    - Place video on YouTube, place link in PDF

Validation

- Make sure program build and run without crashing
    - Converter
    - Game Engine
- Submit the data to perforce
    - Any source fbx file and texture that you used
    - Exported data of the model
    - fbxConverter.exe
- Documentation
    - Write up document
    - Movie recording

Hints

- Do this assignment by iterating and slowly growing your project
    - You won’t be able to finish this assignment in one day - Start now

Troubleshooting

- Focus Input format
    - Get a input format working first
    - Make sure you can load and display the format
    - Modify and change the pyramid as an example

- Focus on converter
  - Create data into an internal temporary format
    - Suggestion STL format, Print to a file
  - Next convert the data to your input format (tested from step 1)
- Extra step that REALLY helps
  - Create a standalone program that reads the input binary format
  - Read and print that to a file
  - This way you can visualize your data.

BABY STEPS!