AE TUTORIAL

ALPHAENGINE V3.08

1. Project Setup

Open Visual studio 2019

Add a new C++ **Empty Project**, call it “MyDemo”.

In Configuration Manager

  Solution platform: Remove x86

  Project Platform: Remove Win32

Add a new file “Main.cpp”

Add “Extern” folder near the solution file.

Copy “AlphaEngine_V3.08” folder into “Extern” folder.

In “`Configuration Properties->VC++ Directories`”

  Include Directories: **$(SolutionDir)Extern\AlphaEngine_V3.08\include**

  Library Directories**: $(SolutionDir)Extern\AlphaEngine_V3.08\lib**

In “`Configuration Properties->Linker->input->additional dependencies`”

  Configuration “**Release**”, platform “x64”: add “**Alpha_Engine.lib**”

  Configuration “**Debug**”, platform “x64”: add “**Alpha_EngineD.lib**”

In “`Configuration Properties->Advanced->Character Set`”

  “**Use Multi-Byte Character Set**”

In “`Configuration Properties->Linker->System->SubSystem`”

  “Windows (/SUBSYSTEM:WINDOWS)”

Execute Step2: (Optional) Project Setup: Quality of life

or

Copy your **Alpha_Engine.dll**, **Alpha_EngineD.dll** and **freetype.dll** to the execution path and jump to step 3: Code Start

2. (optional) Project Setup: Quality of life

On the disk, Remove the garbage folders in **Solution folder**:

Debug, Release, x64

On the disk, Remove the garbage folders in **Project Folder:**

Debug, Release, x64

In “`Configuration Properties->Output Directory`”:

$(SolutionDir)\bin\$(Configuration)-$(Platform)\

In “`Configuration Properties->intermediate Directory`”:

$(SolutionDir)\.tmp\$(Configuration)-$(Platform)\

In “`Configuration Properties->Debugging- Working Directory`”:

$(SolutionDir)\bin\

Copy **Alpha_Engine.dll**, **Alpha_EngineD.dll** and **freetype.dll** and resource files (textures, fonts) in the **bin** folder of your solution

3. Code Start

Copy/Paste/replace the code from “Code01.cpp” file to your main.cpp.
Build -> Run -> will give an empty window with console window. (Title is changed).

4. Red Triangle (mesh/only color)

Under // Variable declaration, add the following variable:

```
AEGfxVertexList * pMesh1 = 0;                          // Pointer to Mesh (Model)
```

Under // Creating the objects (Shapes), add the code in “code02.cpp”.

Under // Game loop draw, add the code in “code03.cpp”

After the game loop ends, add the following code:

```
// Freeing the objects and textures
AEGfxMeshFree(pMesh1);
```

Run the game to see the RED triangle.

5. Change the background color

In the // Initialization section add the following code:

```
//set background color
AEGfxSetBackgroundColor(0.0f, 0.0f, 0.0f);
```

6. Add 2 triangles (form a rectangle) – First only colored.

Under // Variable declaration, add the following variable:

```
AEGfxVertexList * pMesh2 = 0;                          // Pointer to Mesh (Model)
```

Under // Creating the objects (Shapes), add the code in “code04.cpp”.

Under // Game loop draw, add the code in “code05.cpp”

After the game loop ends, add the following code:

```
AEGfxMeshFree(pMesh2);
```

7. Draw another pMesh2 object (object 3):

Under // Game loop draw, add the code in “code06.cpp”

8. Add a blue tint color to object 3:

Under // Game loop draw, add the following:

For pMesh1:

```
    // No tint
    AEGfxSetTintColor(1.0f, 1.0f, 1.0f, 1.0f);
```

For pMesh2 – Object2:

```
// No tint
    AEGfxSetTintColor(1.0f, 1.0f, 1.0f, 1.0f);
```

For pMesh2 – Object3:

```
    // Add Blue tint
    AEGfxSetTintColor(0.0f, 0.0f, 1.0f, 1.0f);
```

This will add a blue color on top of the rectangular pMesh2 – Object2 only.

9. Using Textures

Add the following under // Variable declaration

```
AEGfxTexture * pTex1;                                    // Pointer to Texture (Image)
```

Load a texture (pTex1) from a file.

Under // Loading textures (images) add the following code:

```
// Texture 1: From file
pTex1 = AEGfxTextureLoad("PlanetTexture.png");
AE_ASSERT_MESG(pTex1, "Failed to create texture1!!");
```

Notes: Copy “PlanetTexture.png” file and paste it in the project’s folder.

Under // Freeing the objects and textures write the following:

```
AEGfxTextureUnload(pTex1);
```

Use the pTex1 in the drawing of both pMesh2 objects (2 and 3):

Update the UVs coordinates of both triangles of pMesh2 as in Code07.cpp file.

Update the drawing code of pMesh2 – (object2 and object3) as in Code08.cpp file.

10. Set Blending Mode:

Under “Game Loop Update” section, add the code in “Code09.cpp” file.

Add the following code to both object2 and object3:

```
// Set Transparency
AEGfxSetTransparency(1.0f);
```

Transparency will work only if the Blend mode is not NONE.

11. Move object 1:

Add the following code, under “Variable declaration” section, to declare the positioning variable:

```
float obj1X = 0.0f, obj1Y = 0.0f;            // Position variables for object 1
```

We will use these variables to move object 1:

Under “Game Loop Update” section add the code in “Code10.cpp” file. It uses the arrow keys.

Under “Game Loop Draw” section, update the code of “Drawing object 1” to the following:

```
// Set position for object 1
AEGfxSetPosition(obj1X, obj1Y);
```

12. Playing with textures offsets:

Add the following code, under “Variable declaration” section, to declare the textures offset variables:

```
float objtexX = 0, objtexY = 0;        // Texture variables for object 2 and 3 texture
```

Under “Game loop Update” section add the code in “Code11.cpp” to update the values of the texture offset variables on keyboard input.

For object2 and object3 drawing sections update their texture set functions as follow:

```
// Set texture
AEGfxTextureSet(pTex1, objtexX, objtexY);
```

13. Add a second texture:

Add the following under // Variable declaration

```
AEGfxTexture * pTex2;                                    // Pointer to Texture (Image)
```

Load a texture (pTex2) from a file.

Under // Loading textures (images) add the following code:

```
// Texture 2: From file
pTex2 = AEGfxTextureLoad("YellowTexture.png");
AE_ASSERT_MESG(pTex2, "Failed to create texture2!!");
```

Notes: Copy “YellowTexture.png” file and paste it in the project’s folder.

Under // Freeing the objects and textures write the following:

```
AEGfxTextureUnload(pTex2);
```

14. Applying more than one texture frame per object (simulating animated object)

We will add a variable “counter” to act as frame delays.

```
int counter = 0;                                            // Counter to swap textures
```

We will animate both object 2 and 3. Under “Game loop draw” update their code from “Code12.cpp” file.

15. Drawing lines:

Add a new mesh variable to draw a mesh of lines:

Under “Variable declaration” section, add the following:

```
AEGfxVertexList * pMeshLine;                          // Pointer to Mesh (Model)
```

Under “Creating the objects” section add the following code from “Code13.cpp” file.

Under “Game loop draw” section add the following code:

```
AEGfxSetRenderMode(AE_GFX_RM_COLOR);
AEGfxSetPosition(0.0f, 0.0f);
AEGfxMeshDraw(pMeshLine, AE_GFX_MDM_LINES_STRIP);
```

Under // Freeing the objects and textures write the following:

```
AEGfxMeshFree(pMeshLine);
```

16. Moving the camera:

Add 2 variables to hold the camera x and y coordinates:

```
float camX, camY;                                // Used to temporary store camera position
```

Under “Game loop update” section add the following code from “Code14.cpp”.

17. Display Frame rate using a text (Font)

Add the following font Id variable under //Variable declaration

```
S8 fontId = 0;
```

Add the following under //Creating Fonts

```
fontId = AEGfxCreateFont("Roboto-Regular.ttf", 12);
```

Under “Game loop draw” section add the code from “Code15.cpp”

At the end destroy the font under //Freeing the objects and textures:

```
AEGfxDestroyFont(fontId);
```

Notes: Copy “Roboto-Regular.ttf” file and paste it in the project’s folder.

18. Display the text at the top Right corner

Under the Text draw section add the code from “Code16.cpp”