# Game Engine Dev – Assignment 1

# Part 2

# Adam Andri – 100970401

# Fedor Andreev - 100842078

## Project Standards Agreed Upon By Group

1. Coding Style
   * **Bracing:** All braceswill be placed on new lines and lined up.
   * **Naming Prefixes (examples in parentheses):**
     + Global Variables = G (*g\_Counter)*
     + Members Variables = M (m\_Counter)
     + Pointer Variables = P (m\_pActor)
     + Virtual Functions = V (VDraw())
     + Interface Classes = I (IDrawable)
   * **Variables & Parameters –** All variables will be named using lowercase letters for the first word, followed by capitalizing the first letter in each subsequent compound word word. *Example: g\_Counter*
   * **Functions, Classes, Typedefs & Methods:**

All will be named by capitalizing each compound word within the name. No space will be placed between the indentifier and braces.

*Example: Action\* ThisIsFunctionOne(void);*

* Identical functions will be named the same and given the same signature across all classes.

*Example:*

*Class ClassOne { void OnUpdate(const int deltaMillisconds); }*

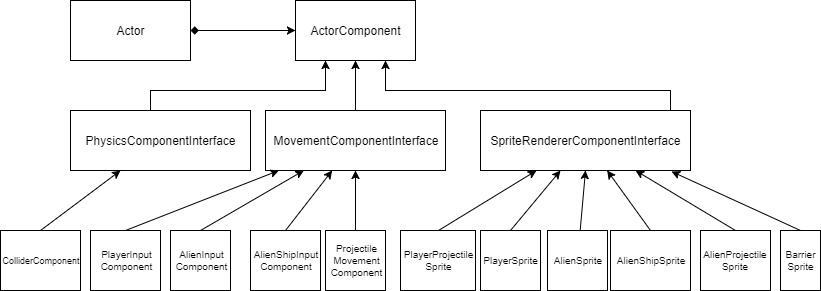
*Class ClassTwo {void OnUpdate(cons tint deltaMillisconds); }*

* + **Macros & Constants:** Names will be all uppercase and compound words will be separated using underscores. *Example: MAX\_SIZE*
  + **Class Layout –** Member variables will be placed at the top, followed by initialization and destruction functions like constructor and destructor. Public interface will be followed by protected and private internal function definitions.
  + **Comments –** Comments will be used whenever code is not easily understandable, such as in the case of a variable that is not named descriptively.

*Example: int a // loop counter*

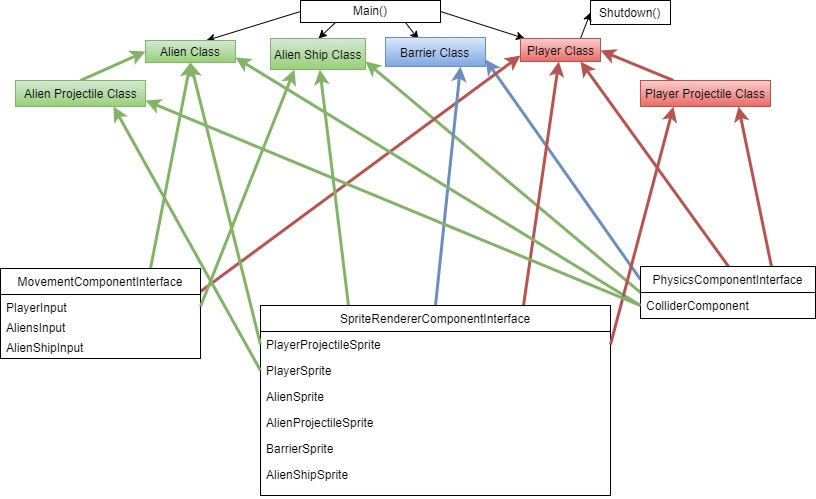
1. Project Structure
   * Docs
   * Assets
   * Source
   * Temp
   * Test
   * Game
   * Lib
2. Minimum PC Requirements
   * Windows 8 or newer
   * 1 GHz 32 bit (x86) or 64 (x64) CPU
   * 512MB RAM
   * DirectX 9.0c
   * DirectX Video Card
   * 100MB Free HD space

## Component Architecture

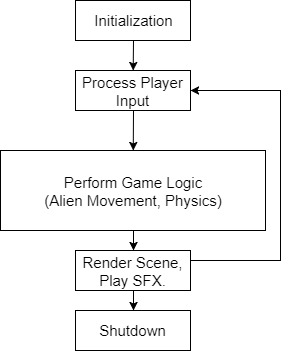
****

* **Actor**
  + **Actor ComponentPl**
    - **PhysicsComponentInterface**
      * **Collider**
    - **MovementComponentInterface**
      * **PlayerInput**
      * **AlienInput**
      * **AlienShipInput**
      * **ProjectileMovement**
    - **SpriteRendererComponentInterface**
      * **PlayerProjectileSprite**
      * **PlayerSprite**
      * **AlienSprite**
      * **AlienShipSprite**
      * **AlienProjectileSprite**
      * **BarrierSprite**

## Class Diagrams



Main Loop



The main loop will update once each frame. The game and all actors will be initialized first before the loop. While the loop continues, all actors will update each frame.

Main()

{

Init(); // Initialize game.

SpawnPlayer(); // Spawns the player with 3 lives.

SpawnAliens(); // Spawns all enemies.

SpawnBarriers() // Spawns barriers.

LoopIsRunning = true; // initializes the loop’s bool condition.

while (LoopIsRunning == true) // while loop is not done running.

{

DrawPlayer(); // while the loop is running, draw the player each frame.

DrawAliens(); // while the loop is running, draw the enemies each frame.

DrawBarriers(); // while the loop is running, draw the barriers each frame.

DrawProjectiles();// while the loop is running, draw the barriers each frame.

If (EscapePushed() == TRUE) // if key is pushed, stop the loop.

loopIsRunning = false;

NextFrame(); // updates to the next frame.

}

Shutdown(); // shutdown the loop when called.

}