**PEMROGAMAN 4**



**MENCARI DAN MENGANALISA**

**SOURCE CODE GAME**

Disusun oleh :

Mochammad Mustakim 4210161003

**PROGRAM STUDI TEKNOLOGI GAME**

**DEPARTEMEN TEKNOLOGI MULTIMEDIA KREATIF**

**POLITEKNIK ELEKTRONIKA NEGERI SURABAYA**

**SURABAYA**

**2018**

1. Class – class pada game

* Class Enemy

#pragma once

#ifndef ENEMY\_H

#define ENEMY\_H

#include "SDLGameObject.h"

class Enemy : public SDLGameObject

{

public:

Enemy(const LoaderParams\* pParams);

virtual void draw();

virtual void update();

virtual void clean();

};

#endif

Class ini berguna untuk membuat dan mengontrol seluruh aktifitas dari enemy. Mulai dari gerakan, tempatnya keluar. Class ini memanggil fungsi yang ada pada Class SDLGameObject.

* Class GameObject

#ifndef GAMEOBJECT\_H

#define GAMEOBJECT\_H

#include <string>

#include "SDL.h"

#include "LoaderParam.h"

class GameObject

{

public:

virtual void draw() = 0;

virtual void update() = 0;

virtual void clean() = 0;

protected:

//GameObject() {};

GameObject(const LoaderParams\* pParams) {}

virtual ~GameObject() {}

};

#endif

Class ini berisi fungsi – fungsi untuk meberikan tampilan pada game. Pada class ini terdapat fungsi untuk mendraw atau mencetak gambar ke layar. Juga terdapat fungsi untuk update event yang terjadi. Event disini bisa berupa posisi dari object. Fungsi clean berguna untuk membersihkan memory yang telah digunakan agar game tidak terlalu berat.

* Class Player

#pragma once

#ifndef PLAYER\_H

#define PLAYER\_H

#include "SDLGameObject.h"

class Player : public SDLGameObject

{

public :

//Player() {};

Player(const LoaderParams\* pParams);

virtual void draw();

virtual void update();

virtual void clean();

};

#endif

Class ini berguna untuk membuat dan mengontrol seluruh aktifitas dari player. Mulai dari gerakan, tempatnya keluar. Class ini memanggil fungsi yang ada pada Class SDLGameObject.

* Class Game

#ifndef \_\_Game\_\_

#define \_\_Game\_\_

#include <SDL.h>

//#include <SDL\_image.h>

#include "TextureManager.h"

#include <iostream>

//#include "Player.h"

//#include "Enemy.h"

#include <vector>

#include "GameObject.h"

class Game {

public:

static Game\* Instance()

{

if (s\_pInstance == 0)

{

s\_pInstance = new Game();

return s\_pInstance;

}

return s\_pInstance;

}

bool init(const char\* title, int xpos, int ypos, int width, int height, int flags);

void handleEvents();

void render();

void update();

void clean();

bool running() { return m\_bRunning; }

SDL\_Renderer\* getRenderer() const { return m\_pRenderer; }

//~Game() {}

private:

bool m\_bRunning;

Game() {};

static Game\* s\_pInstance;

SDL\_Window\* m\_pWindow;

SDL\_Renderer\* m\_pRenderer;

//SDL\_Texture\* m\_pTexture;

//SDL\_Rect m\_sourceRectangle;

//SDL\_Rect m\_destinationRectangle;

//Player m\_player;

std::vector<GameObject\*> m\_gameObjects;

};

typedef Game TheGame;

#endif

Class ini juga bisa dikatakan sebagai class utama pada game ini. Class ini sebagai parent atau induk dari kelas kelas lain. Fungsi fungsi yang ada pada kelas ini adalah fungsi fungsi yang nantinya akan digunakan oleh class lain.

* Class TextureManager

#pragma once

#ifndef \_TextureManage\_

#define \_TextureManage\_

#include "SDL\_image.h"

#include "string.h"

#include "map"

#include "SDL.h"

#include "iostream"

class TextureManager

{

public:

TextureManager() {};

~TextureManager() {};

bool load(std::string fileName, std::string id, SDL\_Renderer\* p\_Render);

void draw(std::string id, int x, int y, int width, int height, SDL\_Renderer\* p\_Render, SDL\_RendererFlip flip = SDL\_FLIP\_NONE);

void drawFrame(std::string id, int x, int y, int width, int height, int currentRaw, int currentFrame, SDL\_Renderer\* p\_Render, SDL\_RendererFlip

flip = SDL\_FLIP\_NONE);

std::map<std::string, SDL\_Texture\*> m\_textureMap;

static TextureManager\* Instance()

{

if (s\_pInstance == 0)

{

s\_pInstance = new TextureManager();

return s\_pInstance;

}

return s\_pInstance;

}

private:

static TextureManager \*s\_pInstance;

};

#endif

Class ini menghandle segala urusan tentang grafik pada game. Pada class ini mengatur grafik berupa gambar yang kemudian akan diampilkan.

* Class LoaderParams

#pragma once

#include <string>

#ifndef LOADERPARAMS\_H

#define LOADERPARAMS\_H

class LoaderParams

{

public:

LoaderParams(int x, int y, int width, int height, std::string textureID) : m\_x(x), m\_y(y), m\_width(width), m\_height(height),m\_textureID(textureID) {}

int getX() const { return m\_x; }

int getY() const { return m\_y; }

int getWidth() const { return m\_width; }

int getHeight() const { return m\_height; }

std::string getTextureID() const { return m\_textureID; }

private:

int m\_x;

int m\_y;

int m\_width;

int m\_height;

std::string m\_textureID;

};

#endif

Class ini adalah class yang sangat penting. Class ini berisi fungsi f ungsi untuk mengatur ukuran atau size pada game. Size yang diatur adalah size object dengan ukuran layar. Class ini dikatakan penting karena class ini adalah agregasi dari class lain. Tanpa adanya class ini, object object yang ada pada game akan tidak teratur ukurannya.

1. Analisis Source code

Pada game tersebut terdapat 5 class, antara lain class texture manager, class game, class player, class game object, dan class enemy. Class enemy berguna untuk membuat sebuah enemy dan mengatur semua hal yang enemy dapat lakukan. Pada class enemy ini saya menjadikannya sebagai header. Di dalam class tersebut berisi perintah untuk menampilkan gambar. Perintah untuk memanggil dan mengatur gambar terletak di class SDLGameObject. Class enemy merupakan turunan dari class SDLGameObject. Kemudian juga terdapat class player yang berguna untuk membuat dan mengatur fungsi yang bisa berjalan pada player. Class player ini merupakan turunan dari class SDLGameObject. Class player mengambil fungsi untuk menggambar, update dan clean dari class SDLGameObject.

Agregasi pada code game ini ada pada class enemy yang dimana pada class enemy memanggil class LoaderParams yang berisi fungsi fungsi untuk menentukan bagaimana cara untuk berjalan atau bergerak. Selain class enemy, class player juga memiliki hubungan agregasi dengan class LoaderParams, dan hal yang diambil atau dipakai adalah sama dengan apa yang dipakai oleh class enemy. Kemudian pada code game ini memiliki hubungan yang saling melengkapi satu sama lainnya. Class Player, Class Enemy, Class SDLGameObject dan Class LoaderParams tidak bisa di pisah. Mereka saling membutuhkan satu sama lain. Class SDLGameObject berguna untuk memberikan fungsi untuk menggambar atau mencetak gambar, update dan clean. Sedangkan Class LoaderParams berguna untuk mengatur control dan pergerakan dari object. Jika tidak ada class LoaderParams maka object tidak dapat bergerak. Begitu juga dengan class SDLGameObject, jika tidak ada class tersebut tidak ada, maka object tidak akan bisa kita lihat karena tidak ada fungsi untuk mencetak gambar.