**Hướng dẫn sử dụng ReadStateXML**

Đầu tiên cài đặt **pugixml** trong NUGET pakage manager

Install-Package pugixml -Version 1.8.5

**Lưu ý: Vì bản này cũng chưa hoàn thiện cần hội ý với Lục Trường Phong để tổ chức lại State Manager và mục đích để giúp đọc State nhanh nên tạm đặt tên class là StateRaw**

Bao gồm class StateRaw :

Class State có 2 thuộc tính là **name** và 1 **mảng** 1 chiều các **REC** tương ứng với state đó.

Khi cần sử dụng thì : khai báo State và gọi hàm

Ví dụ muốn tạo 1 state có tên trong file xml là   
“startrun” và file xml có path là “AladdinSprites.xml”

Ta gọi

StateRaw stateStartRun= StateRaw(“AladdinSprites.xml”,”startrun”);

Sau đó nó sẽ tự độngđọc file xml và lưu tên state là **\_name**=“startrun” (tên state dự định sẽ dùng để truy xuất bằng 1 kiểu dictionary trong statemanager, hội ý sau) và trả vể mảng REC là **\_positions** kèm theo số lượng REC là **\_countPositions**

**Sau đó tùy ý sử dụng bằng các phương thức get tương ứng với từng loại thuộc tính.**

**Trường hợp không mở được file Rar thì t để lại code ở đây:**

**StateRaw.h**

#pragma once

#include <string>

#include <Windows.h>

#include "pugixml.hpp"

using namespace pugi;

class StateRaw

{

private:

RECT\* \_positions;

int \_countPositions;

std::string \_name;

public:

StateRaw();

~StateRaw();

StateRaw(char \*path, std::string stateName);

RECT\* getPositions() const;

int getCountPos() const;

std::string getName() const;

};

**StateRaw.cpp**

#include "StateRaw.h"

StateRaw::StateRaw()

{

}

StateRaw::~StateRaw()

{

}

StateRaw::StateRaw(char\* path, const std::string stateName)

{

\_name = stateName;

xml\_document fileXml;

xml\_parse\_result result = fileXml.load(path);

xml\_node root = fileXml.child("Animations");

for (xml\_node nodeAnimation = root.first\_child(); nodeAnimation; nodeAnimation = nodeAnimation.next\_sibling())

{

if (nodeAnimation.attribute("name").value() == stateName)

{

\_countPositions = 0;

RECT \*rect;

for (xml\_node nodeRec = nodeAnimation.first\_child(); nodeRec; nodeRec = nodeRec.next\_sibling())

{

\_countPositions++;

}

\_positions = new RECT[4];

int i = 0;

for (xml\_node nodeRec = nodeAnimation.first\_child(); nodeRec; nodeRec = nodeRec.next\_sibling(), ++i)

{

\_positions[i].top = LONG(nodeRec.attribute("y").value());

\_positions[i].left = LONG(nodeRec.attribute("x").value());

\_positions[i].bottom = LONG(nodeRec.attribute("h").value()) + \_positions[i].top;

\_positions[i].right = LONG(nodeRec.attribute("w").value()) + \_positions[i].left;

}

}

}

}

RECT\* StateRaw::getPositions() const

{

return \_positions;

}

int StateRaw::getCountPos() const

{

return \_countPositions;

}

std::string StateRaw::getName() const

{

return \_name;

}