จัดทำโดย นายณัชพล ศานติพิบูล รหัสนักศึกษา 60010285

ลองดู(หรือข้ามไปเลยก็ได้...)โค้ดต่อไปนี้ที่ใช้จัดการศัตรูในเกมที่เขียนด้วยวิธีปกติ...

//...

void MoveEnemy(float \*x,float \*y,float ms) {

Movecode(\*x,\*y,ms);

}

void DrawEnemy(float x, float y, float color,int type) {

setDrawColor(color);

if (type == 0)

Draw(x, y, "Enemy1.png");

else (type == 1)

Draw(x, y, "Enemy2.png");

//else if ไปเรื่อยๆตามชนิดศัตรู

}

void Shoot\_Bullet(float x, float y, float damage, float range) {

//Shoot\_Bullet code...

}

void Shoot\_Lazer(float x, float y, float damage, float range) {

//Shoot\_Lazer code...

}

//void Shoot\_xxx(float x, float y, float damage, float range) ไปเรื่อยๆตามชนิดศัตรู

//...

int main() {

//...

//Status ศัตรู

bool Enemy\_active[200];

float Enemy\_x[200];

float Enemy\_y[200];

float Enemy\_HP[200];

int Enemy\_color[200];

int Enemy\_type[200];

float Enemy\_attack\_delay[200];

float Enemy\_attack\_damage[200];

float Enemy\_attack\_speed[200];

float Enemy\_attack\_range[200];

float Enemy\_size[200];

float Enemy\_movespeed[200];

//...

//สร้างศัตรู 1 ตัว

for (int i = 0; i < 200; i++) {

if (Enemy\_active[i] == false) {

Enemy\_active[i] = true;

Enemy\_x[i] = 10;

Enemy\_y[i] = 10;

Enemy\_HP[i] = 100;

Enemy\_color[i] = 1;

Enemy\_type[i] = 0;

Enemy\_attack\_speed[i] = 50;

Enemy\_attack\_delay[i] = Enemy\_attack\_delay[1];

Enemy\_attack\_damage[i] = 35;

Enemy\_attack\_range[i] = 30;

Enemy\_size[i] = 3;

Enemy\_movespeed[i] = 10;

break;

}

}

//...

//Update ศัตรู

for (int i = 0; i < 200; i++)

{

//ถ้าศัตรู Active

if (Enemy\_active[i]) {

MoveEnemy(&Enemy\_x[i], &Enemy\_y[i], Enemy\_movespeed[i]);

//วาดศัตรูที่ x y และสีของศัตรู

DrawEnemy(Enemy\_x[i], Enemy\_y[i], Enemy\_color[i], Enemy\_type[i]);

//ถ้าถึงเวลายิงให้ยิงตามประเภทของศัตรู

if (Enemy\_attack\_delay[i] <= 0) {

if (Enemy\_type[i] == 0) {

Shoot\_Bullet(Enemy\_x[i], Enemy\_y[i], Enemy\_attack\_damage[i], Enemy\_attack\_range[i]);

}

else if (Enemy\_type[i] == 1)

{

Shoot\_Lazer(Enemy\_x[i], Enemy\_y[i], Enemy\_attack\_damage[i], Enemy\_attack\_range[i]);

}

//else if ไปเรื่อยๆตามชนิดศัตรู

Enemy\_attack\_delay[i] = Enemy\_attack\_speed[i]; //กำหนดเวลาดีเลย์

}

Enemy\_attack\_delay[i] -= deltatime; //ลบเวลาดีเลย์ด้วยเวลาที่ผ่านไป

}

}

//...

if (Playerbullet\_hit\_Enemy[i]) { //ถ้ากระสุนโดนศัตรู i

Enemy\_HP[i] -= Playerbullet\_Damage;

if (Enemy\_HP[i]<=0) { //ถ้าเลือดน้อยกว่า 0 ให้ตาย

Enemy\_active[i] = false;

}

}

//...

}

* ข้อมูลและ Function ทุกอย่างถูกรวมไว้ใน main() ทำให้โค้ดดูรก
* เราสามารถทำให้ดูดีและจัดการได้ง่ายด้วย Class
* ลดโค้ดใน main() จาก 60 กว่าเหลือแค่ไม่ถึง 10 บรรทัด !!!
* จำนวนศัตรูถูกจำกัดสูงสุดที่ 200
* เราสามารถสร้างศัตรูได้อย่างไม่จำกัด\*ด้วย Dynamic memory
* \*จนกว่าmemory จะเต็ม

Class

Class เป็นสิ่งสำคัญมี่ใช้ในการเขีบนโปรแกรมแบบ [OOP](https://l.facebook.com/l.php?u=https%3A%2F%2Fwww.google.co.th%2Fsearch%3Fq%3Dobject%2Boriented%2Bprogramming%2Bc%252B%252B&h=ATOOYNFaewlDnLdvdx_aArw-N8Jv1sGrs0hTLx-dZLAgLwYzUsl52yprvqCCt3LWfN6j4u49n8i8dfmDB19Td8DPO5DnIrqa0wBzuJjfeCCpd8Qxet-7mKwdC_9-uWlnjU7ApndldruryQ) ที่จะทำให้โปรแกรมของเราจัดการโค้ดของเราได้ง่ายขึ้น ในที่นี้จะสอนเรื่อง Class เบื้องต้นเท่านั้น สามารถหาข้อมูลเกี่ยวกับ Class อย่างละเอียดได้ใน Google

Data Structure(Struct)

http://www.cplusplus.com/doc/tutorial/structures/

Struct ทำให้เราเก็บข้อมูลหลายข้อมูลเข้าด้วยกันเป็นชนิดเดียว สามารถเขียน Struct ได้ดังนี้

struct ชื่อชนิดของstruct {

ชนิดข้อมูล ชื่อข้อมูล1

ชนิดข้อมูล ชื่อข้อมูล2

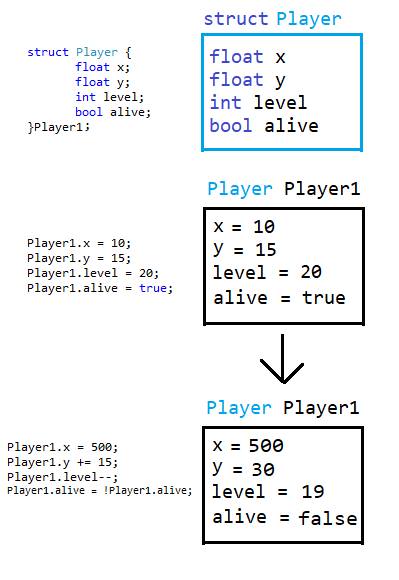
ชนิดข้อมูล ชื่อข้อมูล3

... ...

}ชื่อobject;

โดยเราใช้ ชื่อobject ในการใช้งาน struct นี้ เหมือนกับชื่อตัวแปร และเข้าถึงข้อมูลได้โดยใช้

ชื่อobject.ชื่อข้อมูล



* ซ้าย
* กลาง
* ขวา

ลบออก

Data Structure ทำให้เก็บข้อมูลได้หลายอันในชนิดข้อมูลเดียว

ตัวอย่าง(copy paste ลองรันดูได้) :

#include <iostream>

int main() {

//สร้างstruct Player และobject Player1,Player2

struct Player {

float x;

float y;

int level;

bool alive;

}Player1, Player2; //ใช้ , สร้างหลาย Object เหมือนตัวแปร

//กำหนดค่าต่างๆใน struct

Player1.x = 10;

Player1.y = 15;

Player1.level = 20;

Player1.alive = true;

Player2.x = 10.5f;

Player2.y = -5.5f;

Player2.level = 1;

Player2.alive = false;

//แสดงค่า

printf("Player1 is at (%.1f,%.1f).Level is %d and is %s.\n", Player1.x, Player1.y, Player1.level, Player1.alive ? "alive" : "dead"); //ใช้ ?: แทน if/else http://www.cplusplus.com/articles/1AUq5Di1/

printf("Player2 is at (%.1f,%.1f).Level is %d and is %s.\n", Player2.x, Player2.y, Player2.level, Player2.alive ? "alive" : "dead");

//เปลี่ยนแปลงค่า

Player1.x = 500;

Player1.y += 15;

Player1.level--;

Player1.alive = !Player1.alive;

printf("Player1 is now at (%.1f,%.1f).Level is %d and is %s.\n", Player1.x, Player1.y, Player1.level, Player1.alive ? "alive" : "dead");

Player Player3, Player4; //สร้าง Object เพิ่มได้ เหมือนตัวแปร

Player3.x = 0;

Player3.y = 1;

Player3.level = 2;

Player3.alive = false;

Player4.x = 3;

Player4.y = 4;

Player4.level = 5;

Player4.alive = true;

//แสดงค่า

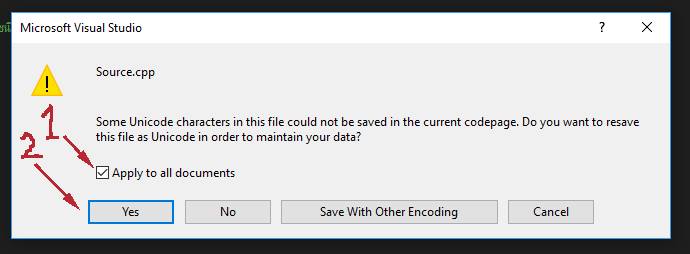
printf("Player3 is at (%.1f,%.1f).Level is %d and is %s.\n", Player3.x, Player3.y, Player3.level, Player3.alive ? "alive" : "dead");

printf("Player4 is at (%.1f,%.1f).Level is %d and is %s.\n", Player4.x, Player4.y, Player4.level, Player4.alive ? "alive" : "dead");

system("pause");

return 0;

}



* ซ้าย
* กลาง
* ขวา

ลบออก

เนื่องจากโค้ดมี Comment ภาษาไทย ทำให้มีปัญหาการ Save ให้กดตามภาพ

Class

http://www.cplusplus.com/doc/tutorial/classes/

Class ก็เหมือนกับ Struct แต่สามารถเก็บ Function ได้ด้วย สามารถเขียน Class ได้ดังนี้

class ชื่อclass {

ชนิดการเข้าถึงข้อมูล1:

สมาชิก1;

ชนิดการเข้าถึงข้อมูล2:

สมาชิก2;

...

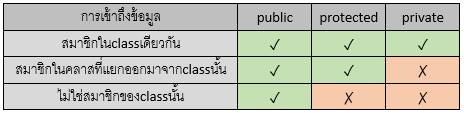
} ชื่อobject;

สมาชิกของ class ก็คือตัวแปร/Function ต่างๆ ที่อยู่ใน Class นั้นๆ

ใช้ ชื่อobject ในการใช้งาน class นี้ เหมือนกับชื่อตัวแปร

แต่ละสมาชิกของคลาสต้องมีชนิดการเข้าถึงข้อมูลกำหนดไว้ โดยมี 3 ประเภทคือ

1. private สมาชิกในคลาสเดียวกันสามารถเข้าถึงข้อมูลได้เท่านั้น ถ้าเราไม่กำหนดชนิดการเข้าถึงข้อมูลไว้จะถือว่าเป็น private โดยอัตโนมัติ
2. protected สมาชิกในคลาสเดียวกันและสมาชิกของคลาสที่แยกมาจากคลาสนั้น(ในที่นี้ไม่ได้สอนไว้)สามารถเข้าถึงข้อมูลได้
3. public สามารถเข้าถึงข้อมูลได้ทุกที่ที่เข้าถึง object นั้นได้



* ซ้าย
* กลาง
* ขวา

ลบออก

ตารางเปรียบเทียบชนิดการเข้าถึงข้อมูล

สามารถเข้าถึงสมาชิกในclassได้เหมือน Struct โดย

ชื่อobject.ชื่อสมาชิก

สามารถสร้าง Function นอกคลาสด้วย Scope Operator (::) ใช้งานโดย

ชื่อclass::ชื่อFunction(){

//ใส่โค้ดของ Function

}

ตัวอย่าง(copy paste ลองรันดูได้) :

#include <iostream>

//สร้างclass Player

class Player {

private: //ใส่ชนิดการเข้าถึงตัวแปร private กับตัวแปรและ Function ที่ไม่อยากให้ถูกเปลี่ยนค่าได้โดยตรง

bool alive;

public:

float x;

float y;

int level;

void die() { //ตายและลดเลเวล

alive = false;

level--;

}

void set\_position(float newx, float newy); //ตั้งค่าตำแหน่ง

void spawn() { //เกิดใหม่และย้ายไปจุดเกิด

alive = true;

set\_position(0, 0);

}

bool get\_alive\_status() { //ถึงแม้ว่า alive เป็น private แต่ก็สามารถใช้ public Function ส่งค่าออกไปได้

return alive;

}

float distance\_from\_spawn() { //ระยะห่างจากจุดเกิด (0,0)

return sqrt((x\*x)+(y\*y));

}

};

void Player::set\_position(float newx, float newy) { //สร้าง Function นอกคลาสด้วย Scope Operator (::)

x = newx;

y = newy;

}

int main() {

Player Player1, Player2; //ใช้ , สร้างหลาย Object เหมือนตัวแปร

Player1.spawn();

Player1.set\_position(10, 15);

Player1.level = 20;

Player2.spawn();

Player2.set\_position(1.5f,2.0f);

Player2.level = 1;

//แสดงค่า

printf("Player1 is at (%.1f,%.1f).Level is %d and is %s.Distance from spawn is %.2f.\n", Player1.x, Player1.y, Player1.level, Player1.get\_alive\_status() ? "alive" : "dead", Player1.distance\_from\_spawn());

printf("Player2 is at (%.1f,%.1f).Level is %d and is %s.Distance from spawn is %.2f.\n", Player2.x, Player2.y, Player2.level, Player2.get\_alive\_status() ? "alive" : "dead", Player2.distance\_from\_spawn());

//เปลี่ยนแปลงค่า

Player1.die();

printf("Player1 is at (%.1f,%.1f).Level is %d and is now %s.\n", Player1.x, Player1.y, Player1.level, Player1.get\_alive\_status() ? "alive" : "dead");

Player1.spawn();

printf("Player1 is now at (%.1f,%.1f) and is %s.\n", Player1.x, Player1.y, Player1.get\_alive\_status() ? "alive" : "dead");

system("pause");

return 0;

}

Class Constructors

Coming soon ™

การนำไปใช้

คราวนี้เราก็สามารถสร้าง Class Enemy จากตัวอย่างโค้ดข้างบนสุด ได้ดังนี้

class Enemy {

public:

bool active;

float x;

float y;

float HP;

int color;

int type;

float attack\_delay;

float attack\_damage;

float attack\_speed;

float attack\_range;

float size;

float movespeed;

Enemy() { //ยังไม่ได้กำหนดตัวแปร

active = false;

}

Enemy(float Enemy\_x, float Enemy\_y, float Enemy\_HP, int Enemy\_color, int Enemy\_type, float Enemy\_attack\_damage, float Enemy\_attack\_speed, float Enemy\_attack\_range, float Enemy\_size, float Enemy\_movespeed) {

active = true;

x = Enemy\_x;

y = Enemy\_y;

HP = Enemy\_HP;

color = Enemy\_color;

type = Enemy\_type;

attack\_delay = Enemy\_attack\_speed;

attack\_damage = Enemy\_attack\_damage;

attack\_speed = Enemy\_attack\_speed;

attack\_range = Enemy\_attack\_range;

size = Enemy\_size;

movespeed = Enemy\_movespeed;

}

void MoveEnemy() {

Movecode(x, y, movespeed);

}

void DrawEnemy() {

setDrawColor(color);

if (type == 0)

Draw(x, y, "Enemy1.png");

else (type == 1)

Draw(x, y, "Enemy2.png");

//else if ไปเรื่อยๆตามชนิดศัตรู

}

void Shoot() {

if (type == 0)

//Shoot\_Bullet code...

else (type == 1)

//Shoot\_Lazer code...

//else if ไปเรื่อยๆตามชนิดศัตรู

}

void Update(float deltatime) { //โค้ด Update ศัตรู ที่ใส่ใว้ใน main()

if (active) {

MoveEnemy();

//วาดศัตรูที่ x y และสีของศัตรู

DrawEnemy();

//ถ้าถึงเวลายิงให้ยิงตามประเภทของศัตรู

if (attack\_delay <= 0) {

Shoot();

attack\_delay = attack\_speed[i]; //กำหนดเวลาดีเลย์

}

attack\_delay -= deltatime; //ลบเวลาดีเลย์ด้วยเวลาที่ผ่านไป

}

}

void Hit (float damage) { //ถ้ากระสุนโดนศัตรู i

HP -= damage;

if (HP<=0) { //ถ้าเลือดน้อยกว่า 0 ให้ตาย

active = false;

}

}

};

คราวนี้โค้ดที่เกี่ยวกับศัตรูทั้งหมดมาอยู่ใน class เดียว ทำให้จัดการได้ง่าย

สามารสร้างและอัพเดตศัตรูได้โดย

int main() {

//...

//ข้อมูลศัตรู

Enemy EnemyArray[200];

//สร้างศัตรู

for (int i = 0; i < 200; i++)

if (Enemy\_active[i] == false)

EnemyArray[i] = Enemy(10, 10, 100, 1, 0, 35, 50, 30, 1, 10);

//อัพเดตศัตรู

for (int i = 0; i < 200; i++)

EnemyArray[i].Update(deltatime);

//...

if (Playerbullet\_hit\_Enemy[i]) {

EnemyArray[i].Hit(Playerbullet\_Damage);

}

//...

}

จะเห็นได้ว่าจากโค้ดใน main() ที่มี 60 กว่าบรรทัดเหลือแค่ไม่ถึง 10 บรรทัดเท่านั้น

Dynamic memory

Coming soon ™

http://www.cplusplus.com/doc/tutorial/dynamic/

New

Coming soon ™

Delete

Coming soon ™

Class Pointer

http://www.cplusplus.com/doc/tutorial/classes/

Coming soon ™

std::vector

http://www.cplusplus.com/reference/vector/vector/

Coming soon ™

การนำไปใช้

คราวนี้เราก็สามารถเปลี่ยนโค้ดข้างบนสุด ให้ใช้ Dynamic memory ได้ดังนี้

แก้โค้ดใน class Enemy

//ตัวแปร active ไม่จำเป็นอีกต่อไป

class Enemy {

public:

float x;

float y;

float HP;

int color;

int type;

float attack\_delay;

float attack\_damage;

float attack\_speed;

float attack\_range;

float size;

float movespeed;

Enemy(float Enemy\_x, float Enemy\_y, float Enemy\_HP, int Enemy\_color, int Enemy\_type, float Enemy\_attack\_damage, float Enemy\_attack\_speed, float Enemy\_attack\_range, float Enemy\_size, float Enemy\_movespeed) {

x = Enemy\_x;

y = Enemy\_y;

HP = Enemy\_HP;

color = Enemy\_color;

type = Enemy\_type;

attack\_delay = Enemy\_attack\_speed;

attack\_damage = Enemy\_attack\_damage;

attack\_speed = Enemy\_attack\_speed;

attack\_range = Enemy\_attack\_range;

size = Enemy\_size;

movespeed = Enemy\_movespeed;

}

void MoveEnemy() {

Movecode(x, y, movespeed);

}

void DrawEnemy() {

setDrawColor(color);

if (type == 0)

Draw(x, y, "Enemy1.png");

else (type == 1)

Draw(x, y, "Enemy2.png");

}

void Shoot() {

if (type == 0)

//Shoot\_Bullet code...

else (type == 1)

//Shoot\_Lazer code...

}

void Update(float deltatime) {

MoveEnemy();

DrawEnemy();

if (attack\_delay <= 0) {

Shoot();

attack\_delay = attack\_speed[i];

}

attack\_delay -= deltatime;

}

//ถ้าศัตรูตายให้ return true เพื่อจะได้ลบ Object นั้นใน main()

bool Hit (float damage) {

HP -= damage;

if (HP<=0) { //ถ้าเลือดน้อยกว่า 0 ให้ตาย

active = false;

return = true;

}else{

return false;

}

}

};

แก้โค้ดใน main()

int main() {

//...

//ข้อมูลศัตรู

std::vector<Enemy\*> EnemyVector;

//สร้างศัตรู

EnemyVector.push\_back(new Enemy(10, 10, 100, 1, 0, 35, 50, 30, 1, 10));

//อัพเดตศัตรู

for (unsigned int i = 0; i < EnemyVector.size; i++)//ใช้ Unsigned Int เพราะเก็บข้อมูลได้มากกว่า

EnemyVector[i]->Update(deltatime);

//...

if (Playerbullet\_hit\_Enemy[i]) {

if (EnemyVector[i]->Hit(Playerbullet\_Damage)) {

delete EnemyVector[i];

EnemyVector.erase(EnemyVector.begin() + i);

}

}

//...

}