

《现代C++编程实现》电子版: <https://leanpub.com/c01>

基于ChGL的Pong游戏

YouTube频道: [hwdong](#)

博客: hwdong-net.github.io

游戏的程序框架

- 游戏初始化
- 游戏的循环：
 - 处理用户输入
 - 更新游戏的数据
 - 绘制场景

```
int main(){↓  
    //1.初始化↓  
    init();↓  
    ↓  
    //2.游戏循环↓  
    while(running){↓  
        processInput();    //2.1 处理用户输入↓  
        update();           //2.2 更新游戏数据↓  
        renderScene();      //2.3 绘制场景↓  
    }    ↓  
    return 0;↓  
}←
```

用ChGL和函数重写Pong游戏

初始化游戏数据

- 游戏中的数据

```
int ball_x, ball_y, ball_vec_x{0}, ball_vec_y{ 0 }; //球位置和速度←  
int paddle_w, paddle_h;    //挡板的长宽←  
int paddle1_x, paddle1_y, paddle1_vec{0}; //左挡板位置和速度←  
int paddle2_x, paddle2_y, paddle2_vec{0};    //右挡板位置和速度←  
int score1{ 0 }, score2{0}, score1_x, score1_y, score2_x, score2_y; //得分及得分的显示位置
```

```

bool init(const int window_width=100,const int window_height=40){
    if (!initWindow(window_width, window_height)) { // 初始化窗口
        return false;
    }
    ball_x = window_width / 2;
    ball_y = window_height / 2;

    paddle_w = 4; paddle_h = 10;
    paddle1_x = 0; paddle1_y = window_height / 2 - paddle_h / 2;
    paddle2_x = window_width - paddle_w; paddle2_y = paddle1_y;
    paddle1_vec = 3; paddle2_vec = 3;

    score1 = 0; score2 = 0;
    score1_x = paddle_w + 8; score1_y = 2;
    score2_x = window_width - 8 - paddle_w; score2_y=2;
    // auto pong_circle_r2{ 40 }; //禁区半径的平方

    srand((unsigned)time(0)); //生成随机数种子
    random_ball(window_width, window_height);
    return true;
}

```

//初始化球的位置和速度↵

```
void random_ball(const int window_width, const int window_height) {↵
```

```
    ball_x = window_width / 2; ball_y = window_height / 2;↵
```

```
    ball_vec_x = rand() % 3 + 1;    //生成一个随机整数表示球的横向速度↵
```

```
    ball_vec_y = rand() % 3 + 1;    //生成一个随机整数表示球的纵向速度↵
```

```
    if (rand() % 2 == 1) ball_vec_x = -ball_vec_x; //速度可以使负数↵
```

```
    if (rand() % 2 == 1) ball_vec_y = -ball_vec_y; //速度可以使负数↵
```

```
}↵
```

绘制背景

背景包括上下墙壁、左右沟渠和中间分割线，可以用一个函数绘制到画布上：

```
void draw_background() {  
    clearWindow();           //清空为背景颜色  
    int &window_width = framebuffer_width,&window_height = framebuffer_height;  
    auto right{ window_width - 1 }, middle{ window_width / 2 };  
    for (int y = 0; y < window_height; y++) {  
        setPixel(0, y, boundary_color);  
        setPixel(middle, y, boundary_color);  
        setPixel(right, y, boundary_color);  
    }  
    int bottom = window_height - 1;  
    for (int x = 0; x < window_width; x++) {  
        setPixel(x,0, wall_color);  
        setPixel(x,bottom,wall_color);  
    }  
}
```

绘制精灵(球和挡板)

//draw_sprites()绘制场景中的精灵：球、挡板和得分。↵

```
void draw_sprites() {    ↵
```

```
    //绘制球↵
```

```
    setPixel(ball_x, ball_y, ball_color);    ↵
```

```
    //绘制左、右挡板↵
```

```
    for (auto y = paddle1_y; y < paddle1_y + paddle_h; y++)↵
```

```
        for (auto x = paddle1_x; x < paddle1_x + paddle_w; x++)↵
```

```
            setPixel(x,y, paddle_color);    ↵
```


绘制精灵(球和挡板)

```
for (auto y = paddle2_y; y < paddle2_y + paddle_h; y++)  
    for (auto x = paddle2_x; x < paddle2_x + paddle_w; x++)  
        setPixel(x,y, paddle_color);  
  
//绘制分数： 分数是一个字符串  
std::string s1{ std::to_string(score1) }, s2{ std::to_string(score2) };  
for (auto i = 0; i < s1.size(); i++)  
    setPixel(score1_x + i, score1_y, s1[i]);  
for (auto i = 0; i < s2.size(); i++)  
    setPixel(score2_x + i, score2_y, s2[i]);  
}
```

绘制场景

```
void gotoxy(int x, int y) {  
    COORD coord = {x, y };  
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);  
}  
  
void hideCursor() {  
    CONSOLE_CURSOR_INFO cursor_info = { 1, 0 };  
    SetConsoleCursorInfo(GetStdHandle(STD_OUTPUT_HANDLE), &cursor_info);  
}  
  
void render_scene() {  
    gotoxy(0, 0); //定位到(0,0)，相当于清空屏幕  
    hideCursor();  
    draw_background(); //在画布上绘制背景  
    draw_sprites(); //在画布上绘制精灵  
    show(); //在屏幕上显示画布内容（场景）  
}
```

测试一下

```
int main() {  
    if (!init()) {  
        std::cout << "初始化窗口失败！ \n";  
        return 1;   
    }  
    render_scene();  
    return 0;  
}
```

事件处理

```
int processInput() {  
    // 处理事件  
    char key;  
    if (_kbhit()) {  
        key = _getch();  
        if (key == 27) return -1;  
        else if ((key == 'w' || key == 'W') && paddle1_y > paddle1_vec)  
            paddle1_y -= paddle1_vec;  
        else if ((key == 's' || key == 'S') && paddle1_y + paddle1_vec + paddle_h < HEIGHT)  
            paddle1_y += paddle1_vec;  
        else if (key == 72 && paddle2_y > paddle2_vec)  
            paddle2_y -= paddle2_vec;  
        else if ((key == 80) && paddle2_y + paddle2_vec + paddle_h < HEIGHT)  
            paddle2_y += paddle2_vec;  
    }  
    return 0;  
}
```

更新游戏状态（数据）

更新球的位置，检测球与墙壁、挡板是否发生碰撞。↵

```
void update() {↵  
    // 2. 更新数据 ↵  
    ball_x += ball_vec_x;↵  
    ball_y += ball_vec_y;↵  
    if (ball_y < 0 || ball_y >= HEIGHT) {↵  
        ball_vec_y = -ball_vec_y; ↵  
        ball_y += ball_vec_y;↵  
    }↵
```

更新游戏状态 (数据)

```
if (ball_x < paddle_w && ball_y >= paddle1_y && ball_y < paddle1_y + paddle_h) {  
    ball_vec_x = -ball_vec_x;  
    score1 += 1;  
}  
else if (ball_x > WIDTH - paddle_w && ball_y >= paddle2_y && ball_y < paddle2_y + paddle_h) {  
    ball_vec_x = -ball_vec_x;  
    score2 += 1;  
}  
bool is_out{ false };  
if (ball_x < 0) { score2 += 1; is_out = true; }  
else if (ball_x >= WIDTH) { score1 += 1; is_out = true; }  
if (is_out) {  
    random_ball();  
}  
}
```

主函数

```
int main() {  
    //1. 初始化数据  
    init();  
    //2. 游戏循环  
    while (true) {  
        if (processInput() < 0) break;  
        update();  
        render_scene();  
    }  
    return 0;  
}
```

完整代码

https://github.com/hwdong-net/cplusplus17/blob/master/code/6.9%20pong_fun.cpp

关注我

博客: hwdong-net.github.io

Youtube频道



hwdong

@hwdong · 5.01K subscribers · 558 videos

博客: <https://hwdong-net.github.io> >

youtube.com/c/4kRealSound and 4 more links