

GG1003

Windows编程

胡学萱

胡学萱

实验6-5 游戏主控类的实现及使用

余庆祥

2220631136

BDP

2022/12/11

**内页写作格式**

**一、实验名称：**要用最简练的语言反映实验的内容。

**二、实验日期：**写明做实验的具体年、月、日及组別。

**三、实验目的：**使用简洁的文字或关键字来敘述，是以怎样的目的作此实验的。

**四、实验环境：**实验的操作系统和软件等。

**五、实验的步骤和方法：**这是实验报告极其重要的内容。这部分要写明经过哪几个步骤。

**六、数据记录和计算：**指从实验中测到的数据以及计算结果。

**七、实验结果或结论：**即根据实验过程中所见到的现象和测得的数据，得出结论。

**八、备注或说明**：可写上实验成功或失败的原因，实验后的心得体会、建议等。

|  |  |
| --- | --- |
| **1、实验名称** | 实验6-5 游戏主控类的实现及使用 |
| **2、实验日期** | **2022/12/11** |
| **3、实验目的** | |
| 1. 理解框架的概念、作用。 2. 理解CGame类的设计，理解CGame是如何将系统时间的推进、渲染环境、鼠标键盘事件、资源管理、精灵及其管理等等内容抽象、封装的。 3. 理解CGame中各属性的作用，理解其中各成员函数的实现；会使用CGame。 | |
| **4、实验环境** | |
| 1、vs2019 | |
| **5、实验的步骤和方法** | |
| 1、基于CGame派生一个子类CMyGame，在其中增加游戏背景、monster、walker、宝箱等精灵。其中monster、walker都是动画精灵类对象。  分析：  子类需要重写Init，在其中加载所有需要的资源；  增加各精灵成员，由精灵管理类创建，但在CMyGame中保留副本，从而控制各精灵的行为。  2、给monster增加移动逻辑，按某直线往返，或按某矩形区域顺/逆时针旋转；增加walker的移动控制。  分析：  重写需要的键盘接口，例如按方向键之后，改变精灵位置；  判断是否碰撞，从而调用精灵管理类对象删除精灵；  巡逻逻辑不受控制，可考虑在Update中实现。  3、继续增加碰撞逻辑，完成游戏Demo。 思考题 将本次实验中的monster设计成一种特别的精灵（有固定移动逻辑），它既不是CAnimationSprite，也不是CSprite，如何做才能在现在框架的基础上实现对此类对象的管理？试着实现它。 | |
| **6、数据记录和计算** | |
| **（代码编写思路及关键代码）**  DemoGame.h代码：  #include "Game.h"  #define STEP 10  enum Dir {  right,  left,  up,  down  };  class CDemoGame : public CGame  {  public:  CDemoGame(int nFps = 0);  ~CDemoGame(void);  void Init();  private:  CSprite\* pBackGround, \*pBox, \*pApple; // 背景、箱子、苹果  CAnimationSprite\* pMonster, \*pPlayer; // 怪兽、玩家    void HandleKeyDown(WPARAM, LPARAM);  void Update(float);  RECT rt;  Dir dir;  };  DemoGame.cpp代码：  #include "DemoGame.h"  CDemoGame::CDemoGame(int nFps) :CGame(nFps)  {  }  CDemoGame::~CDemoGame(void)  {  }  void CDemoGame::Init()  {  // 用资源管理对象加载资源  pResManager->LoadPictureFromFile(L"Background", L"background.bmp");  pResManager->LoadPictureFromFile(L"Box", L"box.bmp");  pResManager->LoadPictureFromFile(L"Apple", L"apple.bmp");  pResManager->LoadPictureFromFile(L"Monster", L"monster.bmp");  pResManager->LoadPictureFromFile(L"Player", L"walker.bmp");  // 背景  pBackGround = pSpriteManager->CreateSprite(L"Background");  pBackGround->SetDestRect(0, 0, rectClient.right, rectClient.bottom);  // 定义矩形  rt = { 400, 300, 800, 700 };  // 箱子  pBox = pSpriteManager->CreateSprite(L"Box", rt.left + (rt.right - rt.left) / 2, rt.top + (rt.bottom - rt.top) / 2);  // 苹果  pApple = pSpriteManager->CreateSprite(L"Apple", rt.left + (rt.right - rt.left) / 2, rt.top + (rt.bottom - rt.top) / 2);  pApple->SetTransparent(true);  pApple->SetVisible(false);  // monster  pMonster = pSpriteManager->CreateAnimationSprite(L"Monster", 4, 5, rt.left, rt.top, 128, 128);  pMonster->SetTextureStartPos(0, pMonster->GetHeight() \* 2);  pMonster->SetTransparent(true);  pMonster->Play();  dir = right;  // player  pPlayer = pSpriteManager->CreateAnimationSprite(L"Player", 4, 5, 10, 100, 32, 54);  pPlayer->SetTransparent(true);  pPlayer->Play();  }  void CDemoGame::HandleKeyDown(WPARAM wParam, LPARAM lParam)  {  switch (wParam)  {  case VK\_LEFT:  pPlayer->SetTextureStartPos(0, pPlayer->GetHeight());  pPlayer->SetPos(pPlayer->GetXPos() - STEP, pPlayer->GetYPos());  break;  case VK\_RIGHT:  pPlayer->SetTextureStartPos(0, pPlayer->GetHeight() \* 2);  pPlayer->SetPos(pPlayer->GetXPos() + STEP, pPlayer->GetYPos());  break;  case VK\_UP:  pPlayer->SetTextureStartPos(0, pPlayer->GetHeight() \* 3);  pPlayer->SetPos(pPlayer->GetXPos(), pPlayer->GetYPos() - STEP);  break;  case VK\_DOWN:  pPlayer->SetTextureStartPos(0, 0);  pPlayer->SetPos(pPlayer->GetXPos(), pPlayer->GetYPos() + STEP);  break;  }  return;  }  // 更新  void CDemoGame::Update(float fDeltaTime)  {  if (pSpriteManager)  {  pSpriteManager->Update(fDeltaTime);  }  switch (dir)  {  case right:  if (pMonster->GetXPos() < rt.right)  {  pMonster->SetPos(pMonster->GetXPos() + 2, pMonster->GetYPos());  }  else {  pMonster->SetTextureStartPos(0, 0);  dir = down;  }  break;  case down:  if (pMonster->GetYPos() < rt.bottom)  {  pMonster->SetPos(pMonster->GetXPos(), pMonster->GetYPos() + 2);  }  else {  pMonster->SetTextureStartPos(0, pMonster->GetHeight());  dir = left;  }  break;  case left:  if (pMonster->GetXPos() > rt.left)  {  pMonster->SetPos(pMonster->GetXPos() - 2, pMonster->GetYPos());  }  else {  pMonster->SetTextureStartPos(0, pMonster->GetHeight() \* 3);  dir = up;  }  break;  case up:  if (pMonster->GetYPos() > rt.top)  {  pMonster->SetPos(pMonster->GetXPos(), pMonster->GetYPos() - 2);  }  else {  pMonster->SetTextureStartPos(0, pMonster->GetHeight() \* 2);  dir = right;  }  break;  }  // 检测碰撞  // 获取精灵的包围盒  RECT rtSprite = \*(pPlayer->GetBoundingBox());  if (pMonster->CollideWith(rtSprite))  {  pSpriteManager->DeleteSprite(pPlayer, true);  }  if (pBox->CollideWith(rtSprite))  {  pSpriteManager->DeleteSprite(pBox, true);  pApple->SetVisible(true);  }  return;  }  main代码:  #pragma comment(lib,"msimg32.lib")  #include "DemoGame.h"  CGame\* pGame;  int APIENTRY WinMain(HINSTANCE hInstance, HINSTANCE hPrevInst, LPSTR lpszCmdLine, int nCmdShow)  {  pGame = new CDemoGame(60);  if (pGame)  {  pGame->Run(hInstance);  delete pGame;  }  return 0;  } | |
| **7、实验结果或结论（**实验结果怎么样？你从这个实验你学会了什么？得出了什么结论？） | |
| **（实验结果截图+文字说明）**    具体细节如录屏所示： | |
| **8、备注或说明**  **）** | |
| **Cgame类把游戏逻辑无关的代码抽离成一个类，减少了开发者的开发时间，把主要时间集中在游戏逻辑上。** | |