

GG1003

Windows编程

李俊琴

李俊琴

实验4-2 双缓冲实验

余庆祥

2220631136

BDP

2022/10/21

**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4-2 双缓冲实验 |
| **2、实验日期** |  |
| **3、实验目的** | |
| 1.掌握定时器的使用方法。  2.了解屏幕显示闪烁的原因，理解双缓冲的工作原理。  3.掌握双缓冲解决屏幕闪烁的基本步骤，能自主编写代码解决闪烁问题。 | |
| **4、实验环境** | |
| 1、vs2019 | |
| **5、实验的步骤和方法** | |
| 1. 使用定时器，实现一个血条变化。红色的血条，底色为绿色，每隔50ms减少一点，直到血量为0，效果参照如下：   <MyHealthPoint.mp4>     1. 在上题基础上扩充代码，把窗口中画满20\*20的圆，设置定时器，每20毫秒刷新一次（即调用InvalidateRect函数使得整个窗口无效）。之后屏幕会有明显的闪烁，截屏很容易就能截取到闪烁画面，如下图所示。使用双缓冲，解决这个闪烁问题。    思考题 在窗口中绘图，多个篮球在做抛物线运动，用定时器定时刷新从而改变篮球位置，这也引起了闪烁。用双缓冲解决闪烁问题。 | |
| **6、数据记录和计算** | |
| **（代码编写思路及关键代码）**  第一题关键代码：  LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)  {  HDC hdc;  PAINTSTRUCT ps;  HBRUSH hBrush1, hBrush2, hOldBrush;  hdc = BeginPaint(hwnd, &ps);  hBrush1 = CreateSolidBrush(RGB(0, 255, 0));  hBrush2 = CreateSolidBrush(RGB(255, 0, 0));  hOldBrush = (HBRUSH)SelectObject(hdc, hBrush1);  static float mul = 1.0;  float base\_width = 200.0;  float step = 0.01;  switch (message)  {  case WM\_CREATE:  SetTimer(hwnd, 1, 50, NULL);  return 0;  case WM\_PAINT:  SelectObject(hdc, hBrush1);  Rectangle(hdc, 50, 50, base\_width + 50, 20 + 50);  SelectObject(hdc, hBrush2);  Rectangle(hdc, 50, 50, base\_width \* mul + 50, 20 + 50);  break;  case WM\_TIMER:  if (mul > 0)  {  mul -= step;  }  else {  mul = 0; // 有时会遇到不可预期的情况：mul < 0;  }  InvalidateRect(hwnd, NULL, true);  return 0;  case WM\_DESTROY:  SelectObject(hdc, hOldBrush);  DeleteObject(hBrush1);  KillTimer(hwnd, 1);  EndPaint(hwnd, &ps);  PostQuitMessage(0);  return 0;  }  return DefWindowProc(hwnd, message, wParam, lParam);  }  第二题关键代码：  LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)  {  HDC hdc, hdcMem;  PAINTSTRUCT ps;  hdc = BeginPaint(hwnd, &ps);  hdcMem = CreateCompatibleDC(hdc);  RECT rt;  GetClientRect(hwnd, &rt);  HBITMAP hBitmap;  hBitmap = CreateCompatibleBitmap(hdc, rt.right - rt.left, rt.bottom - rt.top);  SelectObject(hdcMem, hBitmap);  switch (message)  {  case WM\_CREATE:  SetTimer(hwnd, 1, 20, NULL);  return 0;  case WM\_PAINT:  FillRect(hdcMem, &rt, (HBRUSH)GetStockObject(WHITE\_BRUSH));  for (int i = 0; i < rt.right; i += 20)  {  for (int j = 0; j < rt.bottom; j += 20)  {  Ellipse(hdcMem, i, j, i + 20, j + 20);  }  }  BitBlt(hdc, 0, 0, rt.right, rt.bottom, hdcMem, 0, 0, SRCCOPY);  break;  case WM\_TIMER:  InvalidateRect(hwnd, NULL, true);  return 0;  case WM\_DESTROY:  KillTimer(hwnd, 1);  DeleteObject(hBitmap);  DeleteDC(hdcMem);  EndPaint(hwnd, &ps);  PostQuitMessage(0);  return 0;  }  return DefWindowProc(hwnd, message, wParam, lParam);  }  思考题关键代码：  LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)  {  HDC hdc, hdcMem;  PAINTSTRUCT ps;  hdc = BeginPaint(hwnd, &ps);  hdcMem = CreateCompatibleDC(hdc);  RECT rt;  GetClientRect(hwnd, &rt);  HBITMAP hBitmap;  hBitmap = CreateCompatibleBitmap(hdc, rt.right - rt.left, rt.bottom - rt.top);  SelectObject(hdcMem, hBitmap);  static int x = 0, y = 0;  int p = 70; // 确定一个焦点位置  static int step = 20;  switch (message)  {  case WM\_CREATE:  SetTimer(hwnd, 1, 100, NULL);  return 0;  case WM\_PAINT:  FillRect(hdcMem, &rt, (HBRUSH)GetStockObject(WHITE\_BRUSH));  for (int x = 0, y = 0; x < 250 || y < rt.bottom; x++)  {  y = x \* x / (2 \* p);  if (x % step == 0) {  Ellipse(hdcMem, x - 25, y - 25, x + 25, y + 25);  Arc(hdcMem, x - 55, y - 25, x - 5, y + 25, x - 5, y + 25, x - 15, y - 25);  Arc(hdcMem, x + 5, y - 25, x + 55, y + 25, x + 15, y - 25, x + 20, y + 25);  }  SetPixel(hdcMem, x, y, RGB(255, 0, 0));  }  BitBlt(hdc, 0, 0, rt.right, rt.bottom, hdcMem, 0, 0, SRCCOPY);  break;  case WM\_TIMER:  step++;  if (step > 25) step = 20;  InvalidateRect(hwnd, NULL, true);  return 0;  case WM\_DESTROY:  KillTimer(hwnd, 1);  DeleteObject(hBitmap);  DeleteDC(hdcMem);  EndPaint(hwnd, &ps);  PostQuitMessage(0);  return 0;  }  return DefWindowProc(hwnd, message, wParam, lParam);  } | |
| **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）**  第一题效果图：      第二题效果图：    思考题效果图： | |
| **8、备注或说明**  **）** | |
|  | |