

《坦克大战》游戏编程说明

1. 前置C语言语法复习

条件语句、循环、函数、指针、结构体、联合体、多文件编程等，相关文档请参考课程附带文件 [C语言基础讲义.pdf](#) 和 [C语言提高讲义.pdf](#)；

2. EasyX图形库

1. EasyX简介

EasyX是针对C++风格的图形库，提供了C/C++编程接口，可以用来快速实现图形和游戏编程，官网：<https://easyx.cn/>

2. 安装

1. 下载安装vs2019，选择“C++桌面开发”模块，参考代码是基于VS2019开发；
2. 执行EasyX.exe安装图形库框架
3. 最后，编写一个简单的绘图程序，检测环境是否安装正确

```
#include <easyx.h>           // 引用图形库头文件
#include <conio.h>
int main()
{
    initgraph(640, 480, EW_SHOWCONSOLE);    // 创建绘图窗口，
    大小为 640x480 像素
    circle(200, 200, 100); // 画圆，圆心(200, 200)，半径 100
    _getch();               // 按任意键继续
    closegraph();           // 关闭绘图窗口
    return 0;
}
```

3. EasyX绘图基本概念

1. 颜色

EasyX中预定义了宏常量来表示常用颜色，当绘制对象需要设置颜色时可以直接选取如下宏定义：

常量	值	颜色
-----	-----	-----
BLACK	0x000000	黑
BLUE	0xAA0000	蓝
GREEN	0x00AA00	绿
CYAN	0xAAAA00	青
RED	0x0000AA	红
MAGENTA	0xAA00AA	紫
BROWN	0x0055AA	棕
LIGHTGRAY	0xAAAAAA	浅灰
DARKGRAY	0x555555	深灰
LIGHTBLUE	0xFF5555	亮蓝
LIGHTGREEN	0x55FF55	亮绿
LIGHTCYAN	0xFFFF55	亮青
LIGHTRED	0x5555FF	亮红
LIGHTMAGENTA	0xFF55FF	亮紫
YELLOW	0x55FFFF	黄
WHITE	0xFFFFFF	白

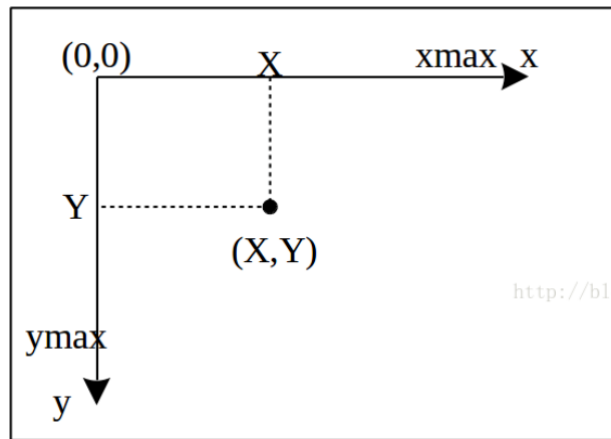
2. 屏幕坐标的概念

1. 在 EasyX 中，坐标分两种：物理坐标和逻辑坐标；

2. 逻辑坐标是在程序中用于绘图的坐标体系，

坐标默认的原点在窗口的左上角，X 轴向右为正，Y 轴向下为正，度量单位是点，

默认情况下，逻辑坐标与物理坐标是一一对应的，一个逻辑点等于一个物理像素；



3. 设备

1. 是指绘图表面，在 EasyX 中，设备分两种，一种是默认的绘图窗口，另一种是 IMAGE 对象；
2. 设置当前用于绘图的设备后，所有的绘图函数都会绘制在该设备上；
3. 绘制之前要先获取绘制的绘图设备句柄（HDC）；

设备句柄HDC可以简单理解为画画用的画布

4. 坦克大战游戏中使用的一些图片绘制函数说明：

1. IMAGE 表示图像对象，用来存储一张图片，详细文档参考<https://docs.easyx.cn/zh-cn/IMAGE>

- IMAGE - 图像对象

```
class IMAGE(int width = 0, int height = 0);
```

- 公有成员

```
int getwidth();
```

返回 IMAGE 对象的宽度，以像素为单位。

```
int getheight();
```

返回 IMAGE 对象的高度，以像素为单位。

```
operator =
```

实现 **IMAGE** 对象的直接赋值。该操作仅拷贝源图像的内容，不拷贝源图像的绘图窗口。

- 示例

以下代码片段创建 **img1**、**img2** 两个对象，之后加载图片 **test.jpg** 到 **img1**，并通过赋值操作将 **img1** 的内容拷贝到 **img2**：

```
IMAGE img1, img2;
loadimage(&img1, _T("test.jpg"));
img2 = img1;
```

以下代码片段创建 **img** 对象，之后加载图片 **test.jpg**，并将图片的宽高赋值给变量 **w**、**h**：

```
IMAGE img;
loadimage(&img, _T("test.jpg"));
int w, h;
w = img.getwidth();
h = img.getheight();
```

2. 绘图常用函数介绍

1. **loadimage()** 加载图片文件到**IMAGE**对象中：<https://docs.easyx.cn/zh-cn/loadimage>

以下代码片段实现了加载“**IMAGE**”分类下的图像资源“**Player**”至 **img** 对象，并在屏幕的指定位置显示：

```
IMAGE img;
loadimage(&img, _T("IMAGE"), _T("Player"));
putimage(100, 100, &img); // 在另一个位置再次显示背景
```

2. **BeginBatchDraw()** 用于开始批量绘图，执行后，任何绘图操作都将暂时不输出到绘图窗口上，直到执行 **FlushBatchDraw** 或 **EndBatchDraw** 才将之前的绘图输出，坦克游戏中使用此种批量绘图：

以下代码实现一个圆从左向右移动，运行发现会有比较明显的闪烁；

然后取消 main 函数中的三个注释语句，打开批绘图功能，对比运动效果，发现批量绘图可以消除闪烁：

```
#include <graphics.h>
int main()
{
    initgraph(640,480);
    // BeginBatchDraw();
    setlinecolor(WHITE);
    setfillcolor(RED);
    for(int i=50; i<600; i++)
    {
        circle(i, 100, 40);
        floodfill(i, 100, WHITE);
        // FlushBatchDraw();
        Sleep(10);
        cleardevice();
    }
    // EndBatchDraw();
    closegraph();
}
```

3. 图像加载到游戏窗口之后，接下来还需要进一步掌握对图像的贴图操作，如下是Windows系统提供的三个常用贴图函数说明：

1. BitBlt()此函数用来拷贝位图资源，不具备缩放能力，主要应用场景：将左上角坐标(x1,y1)的位图资源复制到目的左上角坐标(x,y)宽高分别是cx,cy，不能对原始图像进行缩放

```
// 官方文档: https://docs.microsoft.com/en-us/windows/win32/api/wingdi/nf-wingdi-bitblt
BOOL BitBlt(
    HDC     hdc,
    int     x,
    int     y,
    int     cx,
    int     cy,
    HDC     hdcSrc,
    int     x1,
    int     y1,
    DWORD   rop
);
```

2. **StretchBlt()**, 相比上一个函数**BitBlt()**, 此函数除了可以将一个位图资源从一个矩形区域拷贝到另一个矩形区域, 还具备位图缩放功能

```
// https://docs.microsoft.com/en-us/windows/win32/api/wingdi/nf-wingdi-stretchblt
BOOL StretchBlt(
    HDC     hdcDest,
    int     xDest,
    int     yDest,
    int     wDest,
    int     hDest,
    HDC     hdcSrc,
    int     xSrc,
    int     ySrc,
    int     wSrc,
    int     hSrc,
    DWORD   rop
);
```

3. **TransparentBlt()**, 此函数除了具备上面两个函数的功能位图资源复制和缩放能力之外, 它的最后一个参数可以用来指定是否是透明贴图, 这个透明功能在绘制坦克的时候很有用;

```
// https://docs.microsoft.com/en-us/windows/win32/api/wingdi/nf-wingdi-transparentblt
BOOL TransparentBlt(
```

```

HDC hdcDest,
int xoriginDest,
int yoriginDest,
int wDest,
int hDest,
HDC hdcSrc,
int xoriginSrc,
int yoriginSrc,
int wSrc,
int hSrc,
UINT crTransparent
);

```

4. 基本的图像绘制与贴图功能了解之后，接下来是如何获取按键值控制坦克

1. 获取按键函数原型如下：

```

// 此函数可以在窗口是非焦点的情况下获取到按键值
SHORT GetAsyncKeyState(int vKey);

```

2. 使用示例，检测玩家是否按下了A/a键：

```

if (GetAsyncKeyState('A') & 0x8000) {
    // 检测到按键A的处理代码
}

```

5. 播放游戏音效

1. 加载音效资源

从本地加载名称叫start.wav的音效，并将资源重命名为start

```

mciSendString(_T("open ./res/music/start.wav alias start"),
NULL, 0, NULL);

```

2. 播放调用：

play xxx from 0 表示从头开始播放; play xxx repeat 循环播放; pause xxx 暂停播放









```
mciSendString(_T("play start from 0"), NULL, 0, NULL);
```

6. 游戏定时器的使用

1. 为什么要使用定时器？定时器在坦克大战游戏中非常重要，主要用来更新绘制界面、定时更新坦克坐标、绘制炮弹、绘制闪烁爆炸场景等
2. 使用流程：
 1. 创建一个定时器对象TimeClock
 2. 调用已封装好的clock_init()进行定时器初始化，同时设置定时器时长，单位ms
3. 循环调用clock_is_timeout()查看定时是否完成，完成返回true，否则false
4. 最后根据clock_is_timeout()返回的结果执行定时任务

7. 项目版本

坦克大战游戏由易到难目前一共有8个子版本目录，Github地址为：<https://github.com/gangyahaidao/TankGame>，版本0.1为包含了完整的工程资源文件，其余版本是包含源代码

 TankGame-0.1	2021/1/23 1...	文件夹
 TankGame-0.2	2021/1/23 1...	文件夹
 TankGame-0.3	2021/1/23 1...	文件夹
 TankGame-0.4	2021/1/23 1...	文件夹
 TankGame-0.5	2021/1/23 1...	文件夹
 TankGame-0.6	2021/1/23 1...	文件夹
 TankGame-0.7	2021/1/23 1...	文件夹
 TankGame-0.8	2021/1/23 1...	文件夹

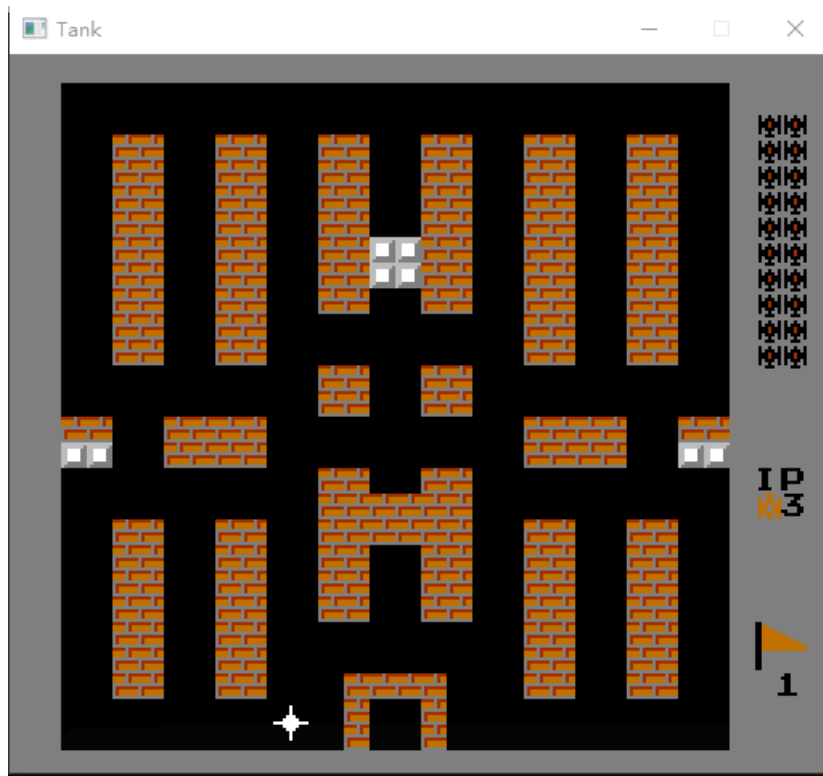
1. 版本0.1：实现游戏模式选择界面



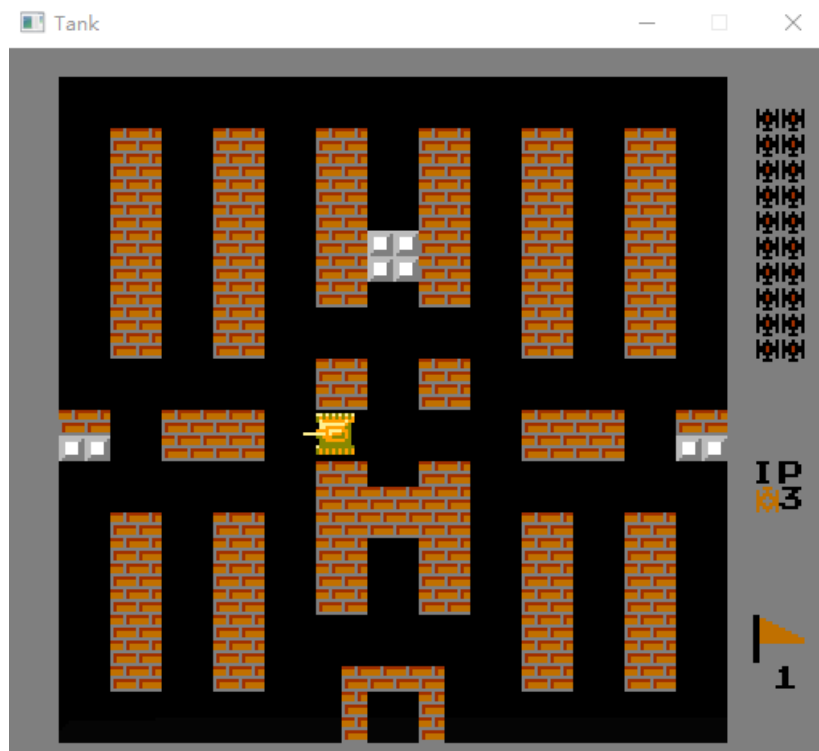
2. 版本0.2: 实现中间游戏区域地图绘制, 以及右侧玩家信息显示



3. 版本0.3: 坦克出生四角星闪烁功能



4. 版本0.4：实现使用键盘ASDW控制玩家坦克移动功能



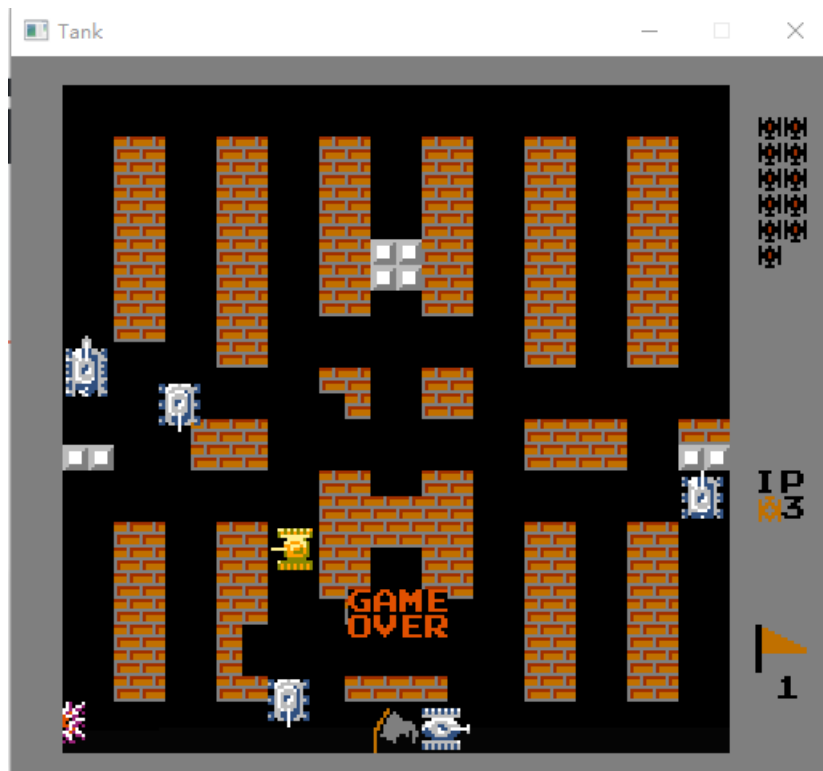
5. 版本0.5：增加玩家坦克移动障碍限制，以及发射炮弹消除砖墙和爆炸特效



6. 版本0.6：实现敌机坦克自主移动和发射炮弹功能，以及玩家消灭敌机功能



7. 版本0.7：增加敌机击毁玩家、玩家与敌机碰撞检测以及敌机击毁大本营显示GameOver游戏循环开始功能；



8. 版本0.8: 增加消灭所有敌机重新开始功能, 并修复之前版本的一些BUG