2018-2019年度第二学期 00106501

计算机图形学

童 伟 华　管理科研楼1205室

E-mail: tongwh@ustc.edu.cn

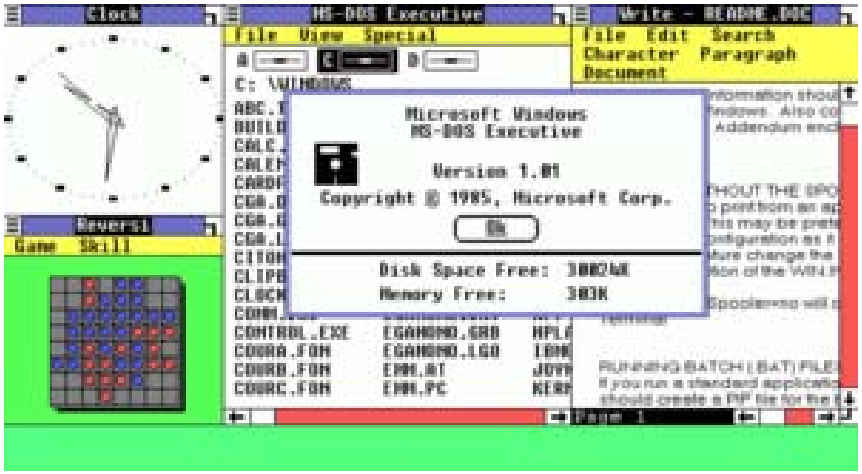
中国科学技术大学　数学科学学院

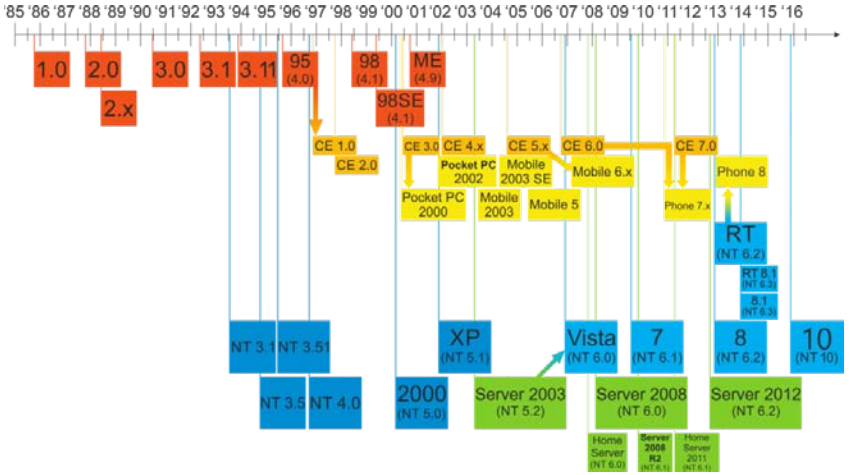
http://math.ustc.edu.cn/

附讲五　Windows编程(API)

Windows操作系统简史

- 1981年，Chase Bishop 提出Interface Manager模型
- 1985年，微软公司发布Windows 1.0版本（运行于DOS操作系统之上）
- 1990年，微软公司发布Windows 3.0版本（运行于多任务DOS操作系统的保护模式之上，引入虚拟内存、虚拟设备驱动器等概念）
- 1992年，微软公司发布Windows 3.1版本



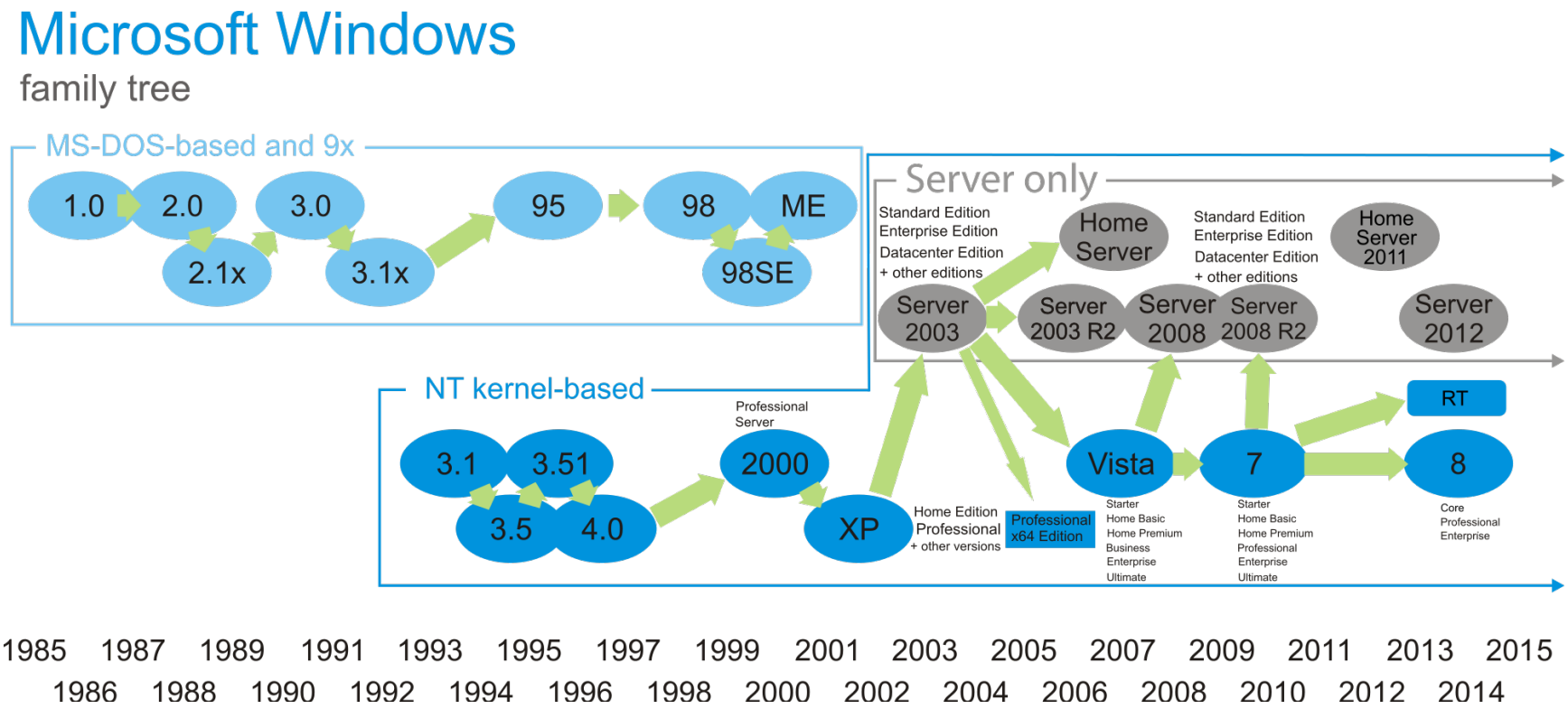


Windows操作系统简史

- 1995年，微软公司发布Windows 95版本（不在运行于DOS操作系统之上，真正的32位操作系统，支持即插即用功能、长文件名等功能）
- 1998年，微软公司发布Windows 98版本（真正取得市场垄断地位的版本）
- 服务器版本：Windows NT版本，例如Windows 2000，Windows Server 2008等
- 2001年，微软公司发布Windows XP版本
- 2009年，微软公司发布Windows 7版本（截至目前，最为成功的版本之一）
- 2012年，微软公司发布Windows 8版本（目标是统一PC，平板，手机等设备的操作系统，不太成功）
- 2015年，微软公司发布Windows 10版本（最新版本）

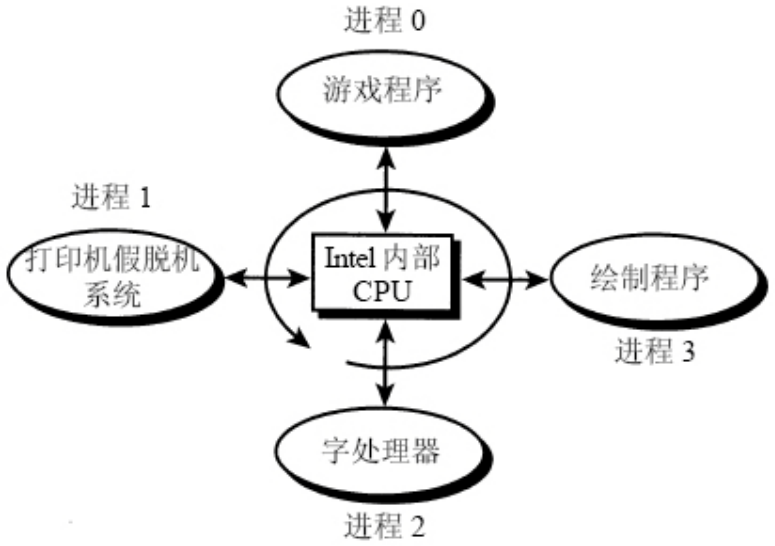
Windows操作系统简史

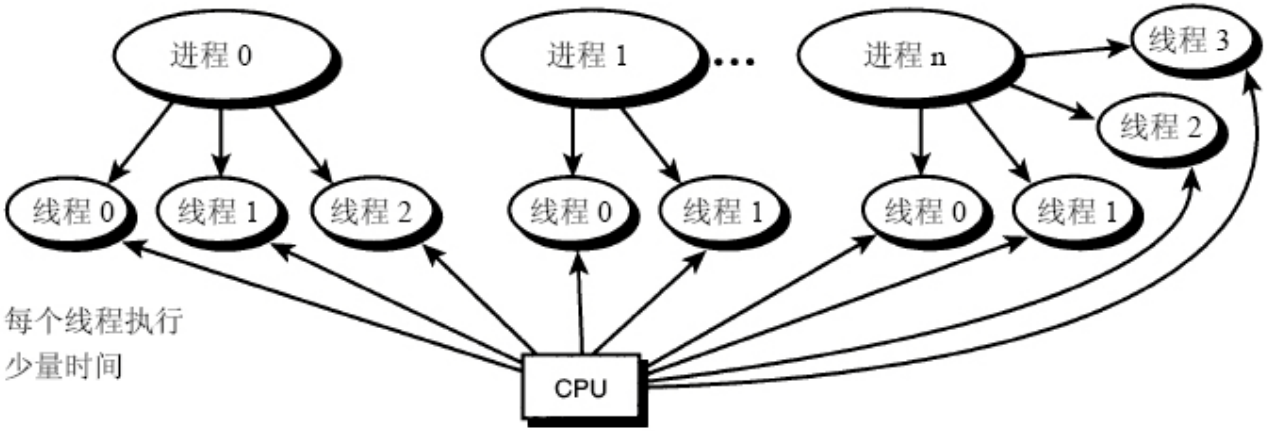
- Windows各版本之间关系图



Windows操作系统特征

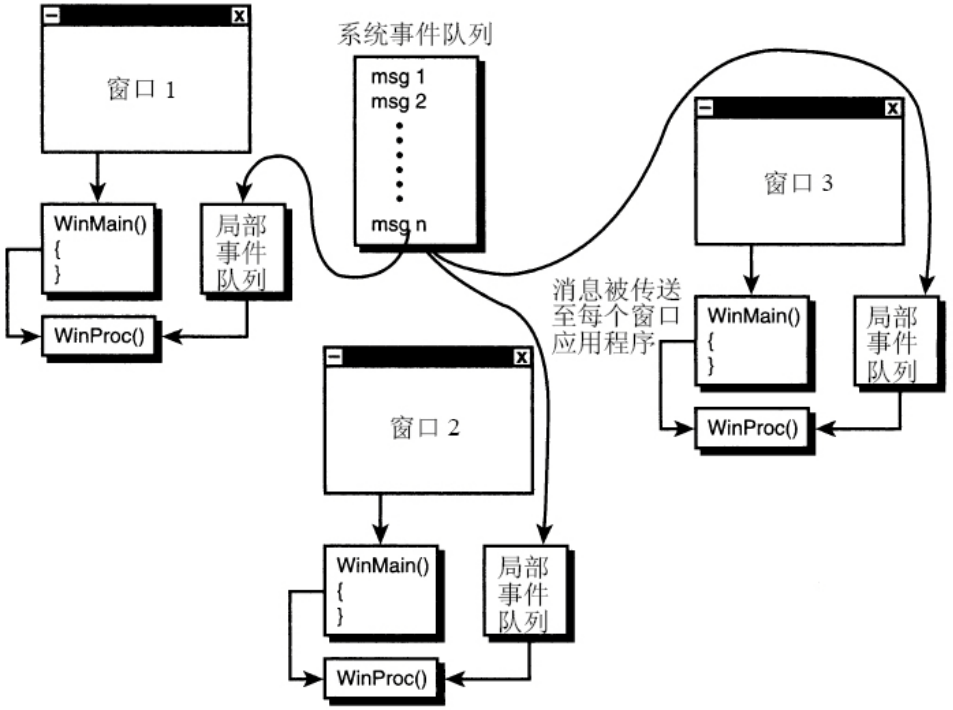
- 简单的，可以用“五多”来刻画：
  - 多用户：多个用户
  - 多任务：多个应用程序
  - 多进程：程序的资源分配单位
  - 多线程：程序的调度单位
  - 多核心：可以有多个处理器





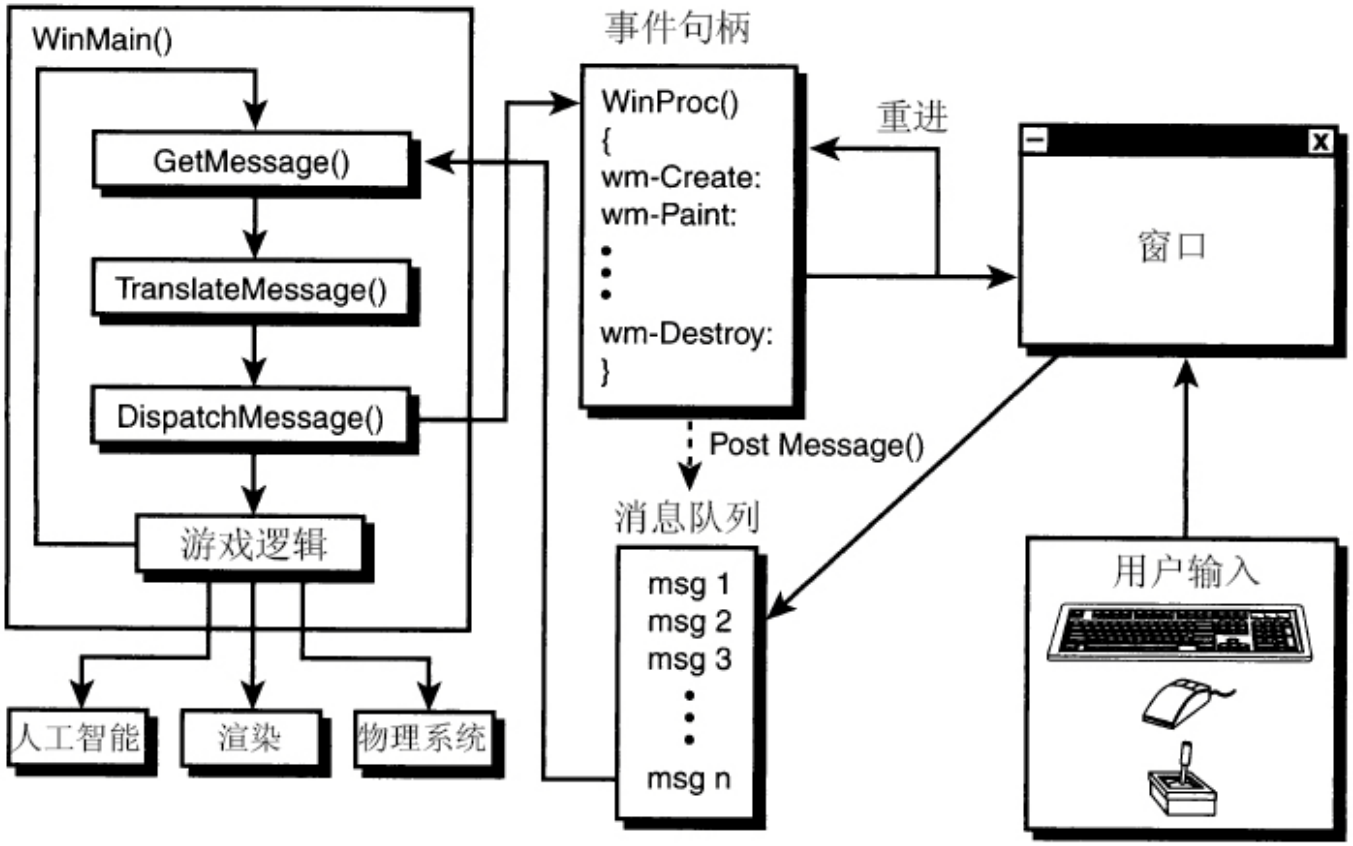
Windows操作系统特征

- **调度方式：基于优先级的抢先占用方式**
  - 一些应用程序要比其他的应用程序占用处理器更多的时间
  - 如果一个应用程序需要CPU处理的话，在另一任务运行的同时，当前的任务可以被锁定或抢先占用
- **驱动方式：事件驱动**



Windows事件模型

- Windows程序事件循环消息机制



Windows编程

- 核心：调用Windows系统提供的API实现编程
- SDK（software development kits）：软件开发包
- 常用方式：
  - Windows API（Application programming interface）编程
  - MFC（Microsoft Foundation Classes）编程
  - .NET编程（WPF）
  - QT编程（目前最流行的图形用户界面应用程序开发框架）

Windows API编程

- 要创建一个简单的Windows程序，需要进行下列工作：
  - 创建一个Windows类
  - 创建一个事件句柄或WinProc
  - 用Windows注册Windows类
  - 用前面创建的Windows类创建一个窗口
  - 创建一个能够从事件句柄获得或向事件句柄传递Windows信息的主事件循环

一个简单的例子

```
// WINMAIN ////////////////////////////////////////////////
int WINAPI WinMain(HINSTANCE hinstance,
                   HINSTANCE hprevinstance,
                   LPSTR lpcmdline,
                   int ncmdshow)
{
   WNDCLASSEX winclass; // this will hold the class we create
   HWND hwnd; // generic window handle
   MSG msg; // generic message

   // first fill in the window class structure
   winclass.cbSize = sizeof(WNDCLASSEX);
   winclass.style = CS_DBLCLKS | CS_OWNDC |
        CS_HREDRAW | CS_VREDRAW;
   winclass.lpfnWndProc = WindowProc;
   winclass.cbClsExtra = 0;
   winclass.cbWndExtra = 0;
   winclass.hInstance = hinstance;
   winclass.hIcon = LoadIcon(NULL, IDI_APPLICATION);
   winclass.hCursor = LoadCursor(NULL, IDC_ARROW);
   winclass.hbrBackground = (HBRUSH)GetStockObject(BLACK_BRUSH);
```

一个简单的例子

```
winclass.lpszMenuName = NULL;
winclass.lpszClassName = WINDOW_CLASS_NAME;
winclass.hIconSm = LoadIcon(NULL, IDI_APPLICATION);

// register the window class
if (!RegisterClassEx(&winclass))
    return(0);

// create the window
if (!(hwnd = CreateWindowEx(NULL, // extended style
    WINDOW_CLASS_NAME, // class
    "Your Basic Window", // title
    WS_OVERLAPPEDWINDOW | WS_VISIBLE,
    0,0, // initial x,y
    400,400, // initial width, height
    NULL, // handle to parent
    NULL, // handle to menu
    hinstance,// instance of this application
    NULL))) // extra creation parms
    return(0);
```

一个简单的例子

```
   // enter main event loop
   while(GetMessage(&msg,NULL,0,0))
   {
        // translate any accelerator keys
        TranslateMessage(&msg);
        // send the message to the window proc
        DispatchMessage(&msg);
   } // end while

   // return to Windows like this
   return(msg.wParam);
} // end WinMain
```

一个简单的例子

```
// GLOBALS ////////////////////////////////////////////////
// FUNCTIONS //////////////////////////////////////////////
LRESULT CALLBACK WindowProc(HWND hwnd,
                            UINT msg,
                            WPARAM wparam,
                            LPARAM lparam)
{
   // this is the main message handler of the system
   PAINTSTRUCT ps; // used in WM_PAINT
   HDC hdc; // handle to a device context
   // what is the message
   switch(msg)
   {
   case WM_CREATE:
        {
                // do initialization stuff here
                // return success
                return(0);
        } break;
```

一个简单的例子

```
   case WM_PAINT:
        {
                // simply validate the window
                hdc = BeginPaint(hwnd,&ps);
                // you would do all your painting here
                EndPaint(hwnd,&ps);
                // return success
                return(0);
        } break;
   case WM_DESTROY:
        {
                // kill the application, this sends a WM_QUIT message
                PostQuitMessage(0);
                // return success
                return(0);
        } break;
   default:break;
   } // end switch

   // process any messages that we didn't take care of
   return (DefWindowProc(hwnd, msg, wparam, lparam));
} // end WinProc
```

Windows API编程

- **推荐书籍：**
  - Charles Petzold，Programming Windows，5th ed., Microsoft Press（权威著作，有中译本，清华大学出版社）
  - Jeffrey Richter，Christophe Nasarre，Windows via C/C++， 5th ed., Microsoft Press（权威著作，有中译本，清华大学出版社）
- **其他资源：**
  - http://en.wikibooks.org/wiki/Windows_Programming
  - http://msdn.microsoft.com/

Thanks for your attention!