Đại học Khoa học Tự nhiên, ĐHQG-HCM
Khoa Công Nghệ Thông Tin

Bài 4: Một số kỹ thuật trong lập trình trên .Net CF

TS. Trần Minh Triết

Xác định đường dẫn của Ứng dụng

- Xác định tự động đường dẫn của ứng dụng (runti

```
m_startuppath =
  System.Reflection.Assembly.GetExecutingAsse
  GetModules()[0].FullyQualifiedName;
m_startuppath = m_startuppath.Replace(
  System.Reflection.Assembly.GetExecutingAssembly().
  GetModules()[0].Name, "" );
m_BmBanCo =new Bitmap
           (m_startuppath+"BanCoPocketPC.jpg");
m_BmQuanCo=new Bitmap
           (m_startuppath+"QuanCoPocKetPC.bmp");
m_BmChonCo=new Bitmap
           (m_startuppath+"ChonQuanPocketPC.bmp");
```

Lấy danh sách các Assembly

Trọn vẹn tên và đường dẫn của module

Xóa tên file, chỉ giữ lại đường dẫn

Tên của module

Xác định đường dẫn của Ứng dụng

- **Sử dụng đường dẫn tuyệt đối (hard-code)!!!**
- **Phải biết trước đường dẫn (tuyệt đối) sẽ chứa chương trình thực thi**

```
public class Constant
{
    public static int LEFT = 24;
    public static int TOP = 24;
    public static string AppPath
                        = @"\Program Files\MummyMaze\";
    public static string ImagesPath
                        = @"\Program Files\MummyMaze\";
}
```

Load ảnh từ file

- Có thể load các ảnh từ file vào đối tượng kiểu Bitmap
- Các định dạng ảnh thông dụng mà WinCE hỗ trợ (BMP, JPG, PNG…)

```
Bitmap RedMummyBmp =
       new Bitmap(Constant.ImagesPath+"redmummy.bmp");

Bitmap HelloBmp =
       new Bitmap(Constant.ImagesPath+"hello.jpg");
```

Sử dụng Timer (1)

- Khai báo biến thuộc kiểu System.Windows.Forms.Timer

```
private System.Windows.Forms.Timer MyTimer;
```

- Khởi tạo biến Timer

```
private void InitializeComponent()
{
    this.MyTimer = new System.Windows.Forms.Timer();
    this.MyTimer.Interval = 300; // 300 ms
    this.MyTimer.Tick +=
          new System.EventHandler(this.MyTimer_Func);
}
```

Tên hàm xử lý Timer

Sử dụng Timer (2)

- **Hàm xử lý mỗi khi xảy ra biến cố timer**

```
private void MyTimer_Func
             (object sender, System.EventArgs e)
{
    flag = 1 - flag;
    pictureBox1.Image = CompleteBmp[flag];
    pictureBox1.Refresh();
}
```

Sử dụng Timer (3)

- Kích hoạt timer

```
MyTimer.Enabled = true;
```

- Tạm dừng timer

```
MyTimer.Enabled = false;
```

- Hủy bỏ timer

```
MyTimer.Dispose();
```

Sử dụng Graphics

```
public void Draw(Graphics g)
{
    ImageAttributes imgatt = new ImageAttributes();
    imgatt.SetColorKey
    (Constant.BkColor, Constant.BkColor);
    g.DrawImage(
    HumanBmp,
    new Rectangle(left, top, width, height),
    Bmp_x*Constant.WidthSquare_pixel,
    Bmp_y*Constant.WidthSquare_pixel,
    Constant.WidthSquare_pixel,
    Constant.WidthSquare_pixel,
    GraphicsUnit.Pixel, imgatt);
}
```

Sử dụng Thread

- Khai báo biến kiểu Thread

```
private Thread SoundThread;
```

- Tạo thread và khởi động thread

```
private void PlaySound()
{   SoundThread =
    new Thread(new ThreadStart(PlaySoundFunc));
    SoundThread.Priority = ThreadPriority.Highest;
    SoundThread.Start(); // Bắt đầu thread
}
```

Tên hàm xử lý chính của Thread

- Hàm xử lý chính của Thread

```
private void PlaySoundFunc()
{   Sound.PlayMusic(Constant.AppPath + "music.wav");
}
```

Xử lý Âm thanh (1)

```
public class Sound
{
    [DllImport("WinMM.dll",
     EntryPoint="PlaySound",CharSet=CharSet.Auto)]
    private static extern int PlaySoundWin32
    (string pszSound, int hmod, int fdwSound) ;

    [DllImport("CoreDll.dll",
     EntryPoint="PlaySound",CharSet=CharSet.Auto)]
    private static extern int PlaySoundWinCE
    (string pszSound, int hmod, int fdwSound) ;
.........
```

Xử lý Âm thanh (2)

```
private enum SND
{
    SND_SYNC          = 0x0000,
    /* play synchronously (default) */
    SND_ASYNC         = 0x0001,
    /* play asynchronously */
    SND_NODEFAULT     = 0x0002,
    /* silence (!default) if sound not found */
    SND_MEMORY        = 0x0004,
    /* pszSound points to a memory file */
    SND_LOOP          = 0x0008,
    /* loop the sound until next sndPlaySound */
```

Xử lý Âm thanh (3)

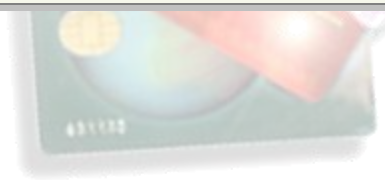
```
private enum SND
{
......
    SND_NOSTOP          = 0x0010,
    /* don't stop any currently playing sound */
    SND_NOWAIT          = 0x00002000,
    /* don't wait if the driver is busy */
    SND_ALIAS           = 0x00010000,
    /* name is a registry alias */
    SND_ALIAS_ID        = 0x00110000,
    /* alias is a predefined ID */
......
}
```

Xử lý Âm thanh (4)

```
private enum SND
{
......
    SND_FILENAME     = 0x00020000,
    /* name is file name */
    SND_RESOURCE     = 0x00040004,
    /* name is resource name or atom */
    SND_PURGE        = 0x0040,
    /* purge non-static events for task */
    SND_APPLICATION     = 0x0080
    /* look for application specific association */
};
```

Xử lý Âm thanh (5)

```
private const int Win32 = 0 ;
private const int WinCE = 1 ;
private static int Windows = -1 ;
public static void PlayMusic(string pszMusic)
{
int flags =
    (int)(SND.SND_ASYNC|SND.SND_FILENAME|
          SND.SND_NOWAIT|SND.SND_LOOP) ;
sndPlaySound(pszMusic, flags) ;
}
```



Xử lý Âm thanh (6)

```
private static void sndPlaySound
    (string pszSound, int flags)
{
    switch ( Windows )
    {
    case Win32 :
        PlaySoundWin32(pszSound, 0, flags) ;
        break ;
    case WinCE :
        PlaySoundWinCE(pszSound, 0, flags) ;
        break ;
    ......
```

Xử lý Âm thanh (7)

```
    default :
    try // Play if in Win32 platform
    {   PlaySoundWin32(pszSound, 0, flags) ;
        Windows = Win32 ;   }
    catch ( Exception )
    {   try // Play if in WinCE platform
        {   PlaySoundWinCE(pszSound, 0, flags) ;
            Windows = WinCE ;   }
        catch ( Exception )
        {
        }
    }
    break ;
    } // switch
}
```