Windows Shell

- A namespace for objects doing files and folders operations
- Sample possibilities:
 - running applications
 - □ drag & drop operations
 - file extensions definitions and their connections with applications
 - □ file icons modifications
 - □ folders appearance modifications
 - CD-ROM autostart
 - creating shortcuts
 - □ using the taskbar and application desktop toolbars
 - using Active Desktop
 - control panel applications

Shell Versions

Libraries:

Comctl32.dll, Shell32.dll, Shlwapi.dll

- differences between versions
- □ included in Windows and Internet Explorer
- version of each library must be checked independently

```
hinstDll = LoadLibrary(lpszDllName);
if (hinstDll) {
    DLLGETVERSIONPROC pGV = NULL;
    pGV = (DLLGETVERSIONPROC)
        GetProcAddress(hinstDll,"DllGetVersion");
    if (pGV) {
        DLLVERSIONINFO dvi;
        ZeroMemory(&dvi, sizeof(dvi));
        (*pGV)(&dvi);
    }
    FreeLibrary(hinstDll);
}
```

Shell Namespace

- One tree structure for all objects managed by the shell
 - e.g.: files, folders, printers, computers in a network, control panel applications, the recycle bin
 - □ elements of the hierarchy:
 - desktop the root
 - system and virtual folders nodes
 - files leaves
- Objects identification
 - □ full path for files (not for virtual files)
 - □ an identifier (item ID) SHITEMID structure
 - list of identifiers ITEMIDLIST structure
 - PIDL pointer to ITEMIDLIST relative or absolute (starting from the desktop)

Folders

- The IMalloc interface memory allocation for PIDL
 - □ SHGetMalloc()
 - IMalloc::Alloc(), IMalloc::Free()
- Special folders identified by constant CSIDL
- IShellFolder interface information about the folder
 content, name, properties
- Functions:
 - □ SHBrowseForFolder() a dialog box to choose a folder
 - □ SHGetPathFromIDList() to get a path for PIDL
 - □ SHGetFolderLocation() to get PIDL for a special folder
 - □ SHGetFolderPath() to get a path for a special folder
 - □ SHGetFileInfo() information about a file
 - SHBindToParent() to get PIDL of an object's parent

Krzysztof Mossakowski Faculty of Mathematics and Information Science

Lecture 13 - 5

Example - SHBrowseForFolder()

```
LPITEMIDLIST pidlRoot = NULL, pidl = NULL;
BROWSEINFO bi = \{0\};
LPMALLOC pMalloc = NULL;
SHGetMalloc(&pMalloc);
SHGetFolderLocation(hWnd, CSIDL PROGRAM FILES,
                    NULL, NULL, &pidlRoot);
bi.hwndOwner = hWnd;
bi.pidlRoot = pidlRoot;
bi.pszDisplayName = buf;
bi.lpszTitle = "Choose a folder";
if ((pidl = SHBrowseForFolder(&bi)) != NULL) {
    SHGetPathFromIDList(pidl, buf);
    pMalloc->Free(pidl);
pMalloc->Free(pidlRoot);
pMalloc->Release();
```

Running Applications

- ShellExecute(), ShellExecuteEx()
 - □ a file or a folder to execute
 - □ an operation to execute
- Operations
 - □ typical:
 - edit run an editor and open a document
 - find start searching from the specified folder
 - open run an application
 - print print a document
 - properties display properties
 - defined for objects in the registry HKEY_CLASSES_ROOT\CLSID\{object_clsid}\Shell\operacja

ShellExecute(hwnd, "find", "C:\\Program Files", NULL, NULL, 0);

File System

- Special features:
 - connections between documents (e.g. all files included in HTML page)
 - □ restrictions for users
 - most recently used documents
- Files operations SHFileOperation()
 - □ copy, move, delete, rename
- Notification about changes SHChangeNotify()
- Adding a document to the list of recently used SHAddToRecentDocs()
- Access to user's folders: CSIDL_MYDOCUMENTS

SHGetFolderLocation(NULL, CSIDL MYDOCUMENTS, NULL, 0, &pidlDocFiles);

Moving Data

- The data
 - □ any object implementing the IDataObject interface
- The clipboard
 - to set the data OleSetClipboard() (the data as an object implementing IDataObject)
 - □ to get the data OleGetClipboard()
 - □ access to the data IDataObject::GetData()
- Drag & drop
 - initialization OleInitialize()
 - □ the data source IDropSource
 - □ the data target IDropTarget

```
HKEY_CLASSES_ROOT
  .zip
     Default)=WinZip
    ContentType =application/x-zip-compressed
PersistentHandler
       (Default)={098f2470-bae0-11cd-b579-08002b30bfeb}
    ShèllEx
       {00021500-0000-0000-c000-00000000046}=
          {E0D79307-84BE-11CE-9641-444553540000}
    ShellNew
       NullFile
  CLSID
    {E0D79306-84BE-11CE-9641-444553540000}
       (Default)=WinZip
InProcServer32
          (Default)=C:\PROGRA~1\WinZip\WZSHLSTB.DLL
          ThreadingModel=Apartment
  WinZip
    DefaultIcon
       (Default)=C:\PROGRA~1\WinZip\winzip32.exe,0
    shell
       open
          [Default]=Open with &WinZip
          command
             (Default)=C:\PROGRA~1\WinZip\winzip32.exe "%1"
    shellex
       DropHandler
          [Default]={E0D79306-84BE-11CE-9641-444553540000}
```

Context Menu

Commands

□ standard commands:

open

print

printto – not visible in context menu

explore - opens Windows Explorer

find - opens Windows Search

openas - opens Open With

properties

custom

HKEY_CLASSES_ROOT .myp (Default)=MyProgram.1 MyProgram.1 (Default)=My Program Application Shell (Default)=doit doit (Default)=&Do it command (Default)=C:\MP\MP.exe /d "%1"

Krzysztof Mossakowski Faculty of Mathematics and Information Science

http://www.mini.pw.edu.pl/~mossakow

Shortcuts

- Shortcuts can be created for files, folders, disks, and printers
- Using the IShellLink interface:
 - □ a place SetPath(), GetPath()
 - □ a working directory SetWorkingDirectory(), GetWorkingDirectory()
 - □ arguments SetArguments(), GetArguments()
 - □ ShowWindow() function parameter SetShowCmd(), GetShowCmd()
 - □ an icon SetIconLocation(), GetIconLocation()
 - □ a description SetDescription(), GetDescription()
 - □ a hot key SetHotKey(), GetHotKey()
 - □ an identifier SetIDList(), GetIDList()

Krz

Fac

Creating a Shortcut

```
HRESULT CreateLink (LPCSTR lpszPathObj,
     LPCSTR lpszPathLink, LPCSTR lpszDesc) {
  HRESULT hres; IShellLink* psl;
  hres = CoCreateInstance(CLSID ShellLink,
             NULL, CLSCTX INPRO\overline{C} SERVER,
             IID IShellLink, (LPVOID*)&psl);
  if (SUCCEEDED(hres)) {
    IPersistFile* ppf;
    psl->SetPath(lpszPathObj);
    psl->SetDescription(lpszDesc);
    hres = psl->QueryInterface(
             &IID IPersistFile, (LPVOID*)&ppf);
    if (SUCCEEDED(hres))
      WCHAR wsz[MAX PATH];
      MultiByteToWideChar(CP ACP, 0,
              lpszPathLink, -1, wsz, MAX PATH);
      hres = ppf->Save(wsz, TRUE);
      ppf->Release();
    psl->Release();
  return hres;
```

Lecture 13 - 12

The Taskbar

The Start menu

- □ before NT 4.0 and 95 use DDE
- since NT 4.0 and 95 use IShellLink and SHGetSpecialFolderLocation() with CSIDL_PROGRAMS
- Quick Launch, Address, Links, Desktop toolbars
 possibility of creating custom bands
- Taskbar buttons
 - □ windows without parents, with WS_EX_APPWINDOW style
 - □ FlashWindow() blinking
- The status area (taskbar notification area aka "tray area")
 icons symbolizing states or events
 - □ Shell_NotifyIcon() adding, modifying, deleting
 - notifications about mouse actions for an icon (left and right click, double click, movement)

Krzysztof Mossakowski Faculty of Mathematics and Information Science

Application Desktop Toolbars

Features:

- □ dockable to the screen's edges
- □ usually with buttons running some actions
- outside the area available for applications

Using:

```
SHAppBarMessage()
```

ABM_NEW, ABM_REMOVE, ABM_SETPOS, ABM_SETAUTOHIDEBAR, ABM_ACTIVATE ABM_QUERYPOS, ABM_WINDOWPOSCHANGED

notifications

ABN_POSCHANGED, ABN_STATECHANGED, ABN_FULLSCREENAPP, ABN_WINDOWARRANGE

Auto Completing

- Automatic hints with endings of texts written in an edit control or ComboBoxEx
- SHAutoComplete() uses auto completing for specified edit control
- Using the IAutoComplete interface
 - one or more sources of hints: history, recently used names, shell's namespace
 - □ two modes
 - autoappend adds a hint to the end of the text
 - autosuggest displays a list with hints

Custom Bands

- Explorer Bar
 - dockable in Internet Explorer
 - □ added to menu View / Explorer Bar
 - □ to create: implement and register a band object
- Tool Band
 - a rebar control with some toolbar controls
 - used in Internet Explorer
- Desk Band
 - a dockable window on the desktop (available from the taskbar's context menu)
- Programming the IDeskBand interface

an entry in the registry: HKEY_LOCAL_MACHINE \ Software \ Microsoft \ Internet Explorer \ Toolbar

Krzysztof Mossakowski Faculty of Mathematics and Information Science

Active Desktop

- Possibility to set HTML documents on the desktop (with ActiveX controls, Java applets, Flash animations, etc.)
- Programming the IActiveDesktop interface
 - □ creating CoCreateInstance() with CLSID_IActiveDesktop
 - □ adding elements AddDesktopItem(), AddDesktopItemWithUI(), AddUrl()
 - □ enumerating elements GetDesktopItemCount(), GetDesktopItem()
 - □ getting and setting a wallpaper GetWallpaper(), SetWallpaper(), GetWallpaperOptions(), SetWallpaperOptions()

Active Desktop – C++ Sample

```
#include "wininet.h"
#include "shlobj.h"
CoInitialize (NULL);
HRESULT hr;
IActiveDesktop *pActiveDesktop;
hr = CoCreateInstance(CLSID ActiveDesktop,
           NULL, CLSCTX INPROC SERVER,
           IID IActiveDesktop,
           (void**) &pActiveDesktop);
pActiveDesktop->SetWallpaper(
           L"C:\\MyWallPaper.gif", 0);
pActiveDesktop->ApplyChanges(AD APPLY ALL);
pActiveDesktop->Release();
```

Windows Programming

Active Desktop – C# Sample

```
IActiveDesktop ad = (IActiveDesktop)new ActiveDesktop();
ad.SetWallpaper(@"C:\MyWallpaper.gif", 0);
ad.ApplyChanges(AD_APPLY.ALL);
```

Windows Themes

- Introduced in the Microsoft Plus! for Windows 95
- Specified in .theme files
 - □ simple text format, similar to .ini files
 - can be created and distributed by independent software vendors
 - □ visible in the Display Properties window
- Windows Themes can modify system settings for a wallpaper, cursors, fonts, sounds, and icons

Visual Styles

- Introduced in Windows XP
- They specify the appearance of controls
 - allow to change the appearance and feel of controls in windows
- UxTheme API to use a particular appearance in an application
- Applied to:
 - non-client parts of windows
 - common controls

🔜 Form1	
Label1	 RadioButton1
Button1	CheckBox1



Krzysztof Mossakowski Faculty of Mathematics and Information Science

ComCtl32.dll and UxTheme.lib

ComCtl32.dll

- included in Windows XP in version 5 (for compatibility) and 6 (for new visual styles)
- □ version 6 includes some new controls
- special manifest must be added to the application to apply visual styles

UxTheme.lib

- a rendering library that separates the visual element of a control from its functionality
- used by the common controls to take advantage of the system's visual styles
- required by ComCtl32.dll

The Manifest

An XML file added as a resource or external file in the application's directory

MANIFEST_ID RT_MANIFEST "YourApp.exe.manifest"

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <assembly xmlns="urn:schemas-microsoft-com:asm.v1"</pre>
      manifestVersion="1.0">
  <assemblyIdentity version="1.0.0.0"</pre>
      processorArchitecture="X86"
      name="CompanyName.ProductName.YourApp"
      type="win32" />
  <description>Your application description here.</description>
  <dependency>
    <dependentAssembly>
      <assemblyIdentity type="win32"</pre>
          name="Microsoft.Windows.Common-Controls"
          version="6.0.0.0"
          processorArchitecture="X86"
          publicKeyToken="6595b64144ccf1df"
          language="*" />
    </dependentAssembly>
  </dependency>
</assembly>
```

Guidelines for Using Visual Styles

- Manifest files will be ignored by previous versions of Windows
- Do not link directly to UxTheme.lib, load the library dynamically
- Write error-handling code to be ready for situations when visual styles do not work as expected
- Test your application to make sure it is not relying on ComCtl32.dll version 6
 - when using ComCtl32.dll version 6, write alternative source code for older systems
 - □ ComCtrl32.dll version 6 is not redistributable

Using UxTheme API

```
HTHEME hTheme = OpenThemeData(hwndButton, "Button");
DrawMyControl(hDC, hwndButton, hTheme, iState);
if (hTheme) {
    CloseTheme(hTheme);
}
void DrawMyControl(HDC hDC, HWND hwndButton, HTHEME hTheme,
   int iState)
{
  if (hTheme) {
    hr = DrawThemeBackground(hTheme, hDC, BP BUTTON,
             iState, &rc, 0);
    hr = GetThemeBackgroundContentRect(hTheme, BP BUTTON,
             iState, &rc, &rcContent);
    hr = DrawThemeText(hTheme, hDC, BP BUTTON,
             iState, szButtonText, cch, DT CENTER
             DT VCENTER | DT SINGLELINE, 0, &rcContent);
  } else {
    // Draw the control without using visual styles
```

Faculty of Mathematics and Information Science

Desktop Window Manager



_

- Desktop Window Manager (DWM) is a new component of Windows Vista
- Through desktop composition, DWM enables visual effects on the desktop as well as various features, such as:
 - □ glass window frames
 - 3-D window transition animations
 - Windows Flip and Windows Flip3D
 - □ high resolution support



Krzysztof Mossakowski Faculty of Mathematics and Information Science

http://www.mini.pw.edu.pl/~mossakow

Sample

Visual Styles in Windows Forms 2.0

- Enabling visual styles
 Application.EnableVisualStyles()
- Controls not affected by visual styles: Label, LinkLabel, DomainUpDown, NumericUpDown, CheckedListBox
- Known issues with controls:
 - □ MonthCalendar selection range set
 - □ TabControl tabs aligned at the bottom or sides
 - TextBox handling of surrogate fonts (pairs of Unicode characters)

Using Visual Styles

[WinForms]

- ControlPaint class rendering common Windows Forms controls
- Classes designed to draw the related control regardless of whether visual styles are available:
 - ButtonRenderer, CheckBoxRenderer, GroupBoxRenderer, RadioButtonRenderer
- Classed designed only to use visual styles:
 - ComboBoxRenderer, ProgressBarRenderer, ScrollBarRenderer, TabRenderer, TextBoxRenderer, TrackBarRenderer

Control Rendering Classes [WinForms]

Visual Style Elements

[WinForms]

Lecture 13 - 30

```
private VisualStyleRenderer renderer = null;
private readonly VisualStyleElement element =
    VisualStyleElement.StartPanel.LogOffButtons.Normal;
public CustomControl()
  if (Application.RenderWithVisualStyles &&
      VisualStyleRenderer.IsElementDefined(element)) {
    renderer = new VisualStyleRenderer(element);
protected override void OnPaint(PaintEventArgs e)
  if (renderer != null) {
    renderer.DrawBackground(e.Graphics,
                            this.ClientRectangle);
  } else {
    this.Text = "Visual styles are disabled.";
    TextRenderer.DrawText(e.Graphics, this.Text,
         this.Font, new Point(0, 0), this.ForeColor);
```

Using a Specific Windows Theme [WPF]

- 1. Add a reference to an appropriate assembly:
 - PresentationFramework.Aero, PresentationFramework.Classic, PresentationFramework.Luna, PresentationFramework.Royale

2. Add a **ResourceDictionary** to the application's resources:

<Application.Resources> <ResourceDictionary> <ResourceDictionary.MergedDictionaries> <!--<ResourceDictionary Source="/PresentationFramework.Royale;component/themes/Royale.NormalColor.xaml" />--> <!--<ResourceDictionary Source="/PresentationFramework.Luna;component/themes/Luna.NormalColor.xaml" />--> <!--<ResourceDictionary Source="/PresentationFramework.Luna;component/themes/Luna.NormalColor.xaml" />--> <!--<ResourceDictionary Source="/PresentationFramework.Luna;component/themes/Luna.Homestead.xaml" />--> <!--<ResourceDictionary Source="/PresentationFramework.Luna;component/themes/Luna.Homestead.xaml" />-->

<ResourceDictionary Source="/PresentationFramework.Aero;component/themes/Aero.NormalColor.xaml" />

<!--<ResourceDictionary Source="/PresentationFramework.Classic;component/themes/Classic.xaml" />--> </ResourceDictionary.MergedDictionaries> </ResourceDictionary> </Application.Resources>

Krzysztof Mossakowski Faculty of Mathematics and Information Science **Windows Programming**

Lecture 13 - 32

Windows Themes Available for WPF

Lu	na.NormalColor.>	caml	Luna.Metallic.xam	nl Aer	ro.NormalColor.xam		
Lue Window1 Label Button CheckBox RadioButton one two three four five TextBox	na.NormalColor.> Window1 Label Button CheckBox RadioButton one two three four five TextBox	kaml Window1	Luna.Metallic.xam	Aer Window1	ro.NormalColor.xam		
					<		
Classic.xaml	assic.xaml Luna.Homestead.xaml Rovale.NormalColor.xaml						

Available WPF Themes

Open source:

- http://wpf.codeplex.com/Wiki/View.aspx?title=WPF%20Themes
- <u>http://www.codeplex.com/wpfthemes</u>





Commercial:

- □ <u>http://www.xamltemplates.net</u>
- □ <u>http://www.nukeation.com/reuxables.aspx?id=catalog</u>

Krzysztof Mossakowski Faculty of Mathematics and Information Science

http://www.mini.pw.edu.pl/~mossakow