# .NETアプリをVista化しよう

中 博俊

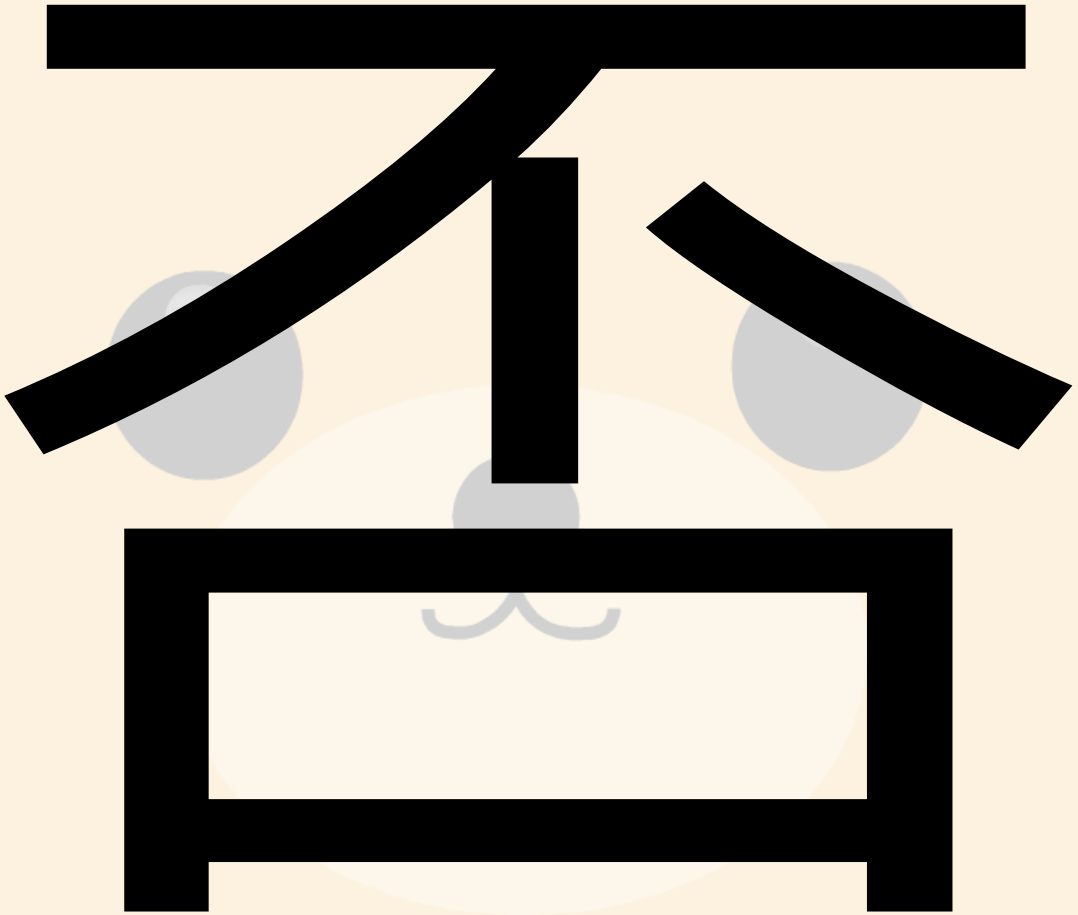
# こんなだいそれた
# タイトルで
# どうしましょう!?

# Windows Vistaの目玉はなんやろか？

Vistaの目玉は何だ

# .NET Framework 3.0 でしょうか？

# Vistaの目玉は何だ

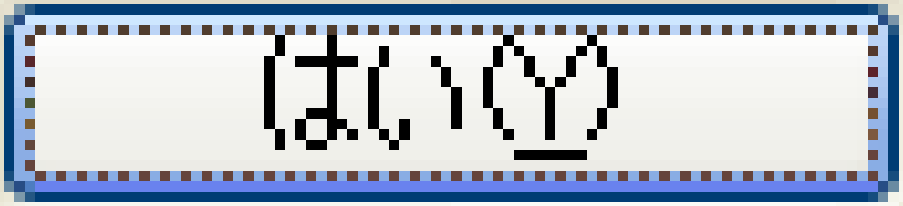


# アプリケーション プラットフォーム としての着実な進化 こそ目玉に相応しい！！

# 今回取り上げる機能は

- ボタン
- IFileDialog
- System.IO.Log(CLFS)
- XPS

# Windows XPのボタンは




# Windows Vistaのボタンは

# DEMO1

違いは明滅
対応するためには
FlatStyleをSystemに
そう.NET 1.1+XPと同じ

# WPFアプリの場合

WPFアプリのエントリポイントはApp.Xamlの

```
<Application x:Class="WindowsApplication1.App"

    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    StartupUri="Window1.xaml"
    >

</Application>
```

# DEMO2

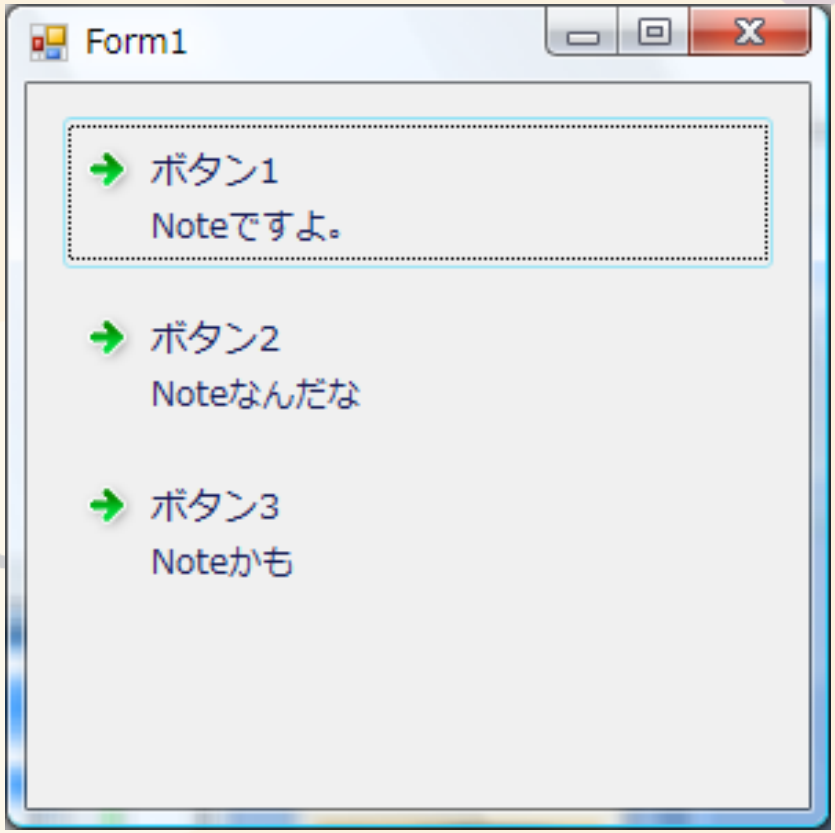
# WPFアプリの場合

WPF + Windows Forms混在アプリを作る場合には
Appクラスのコンストラクタで、EnableVisualStylesする。

```
public partial class App : System.Windows.Application
{
    public App()
    {
        System.Windows.Forms.
        ⇒    Application.EnableVisualStyles();
    }
}
```

# Command Link Button

選択肢を表すにはCommand Link形式のボタンを使いましょう。



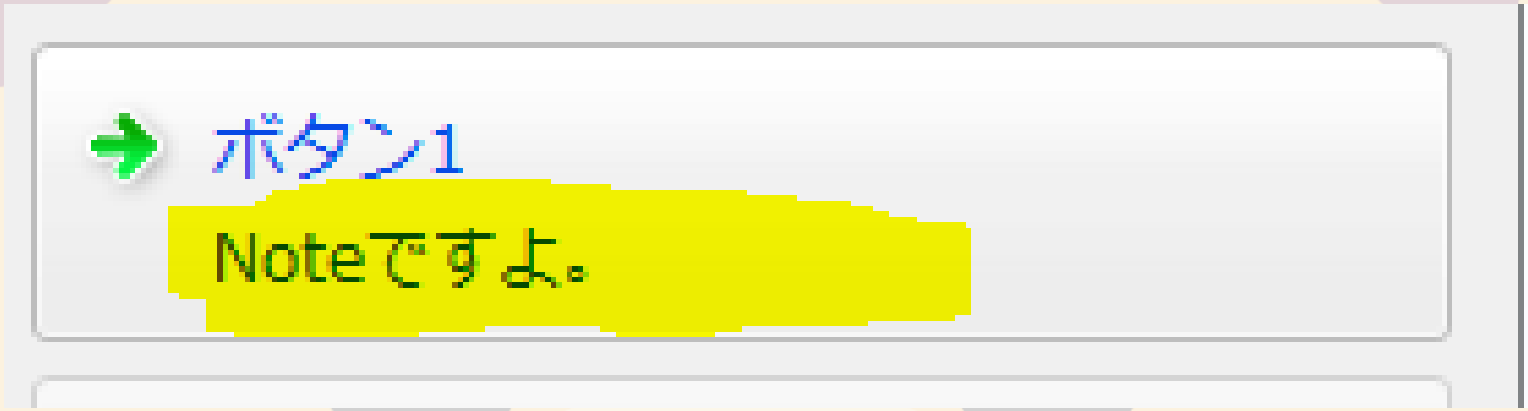
# Command Link Button

実現するにはボタンを継承したコントロールで、CreateParams を上書きしちゃいます。

```
const int BS_COMMANDLINK = 0x0000000E;
protected override CreateParams CreateParams {
    get   {
        if (System.Environment.OSVersion.Version.Major >= 6)   {
                CreateParams cParams = base.CreateParams;
                cParams.Style |= BS_COMMANDLINK;
                return cParams;
        }  else {
                return base.CreateParams;
        }
    } }
```

# Command Link Button

さらにNote部という部分が増えています。



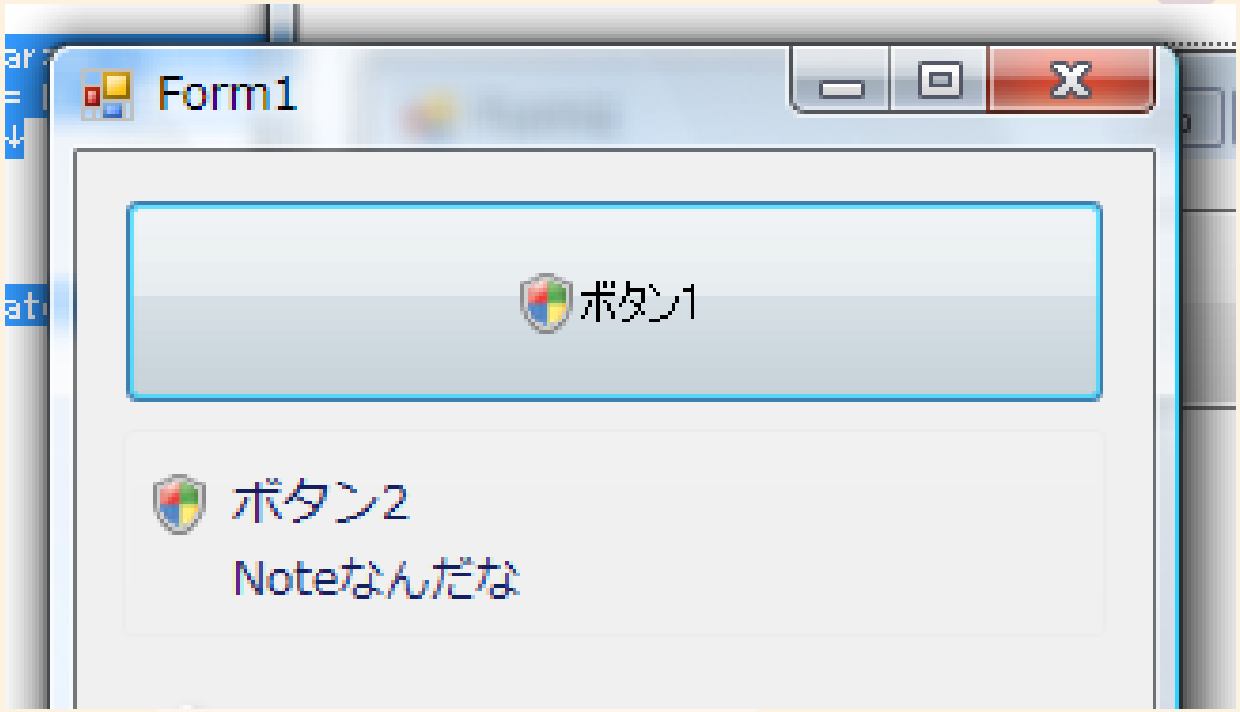

これにはメッセージです。

```
const int BCM_SETNOTE = 0x1609;
SendMessage(new HandleRef(this, this.Handle),
    BCM_SETNOTE, IntPtr.Zero, value);
```

Getは面倒なので割愛

# Shield アイコン

通常のボタンでも、Command Link ButtonでもUACに関係して、昇格が必要な処理のボタンにはシールドアイコンを設定しましょう。



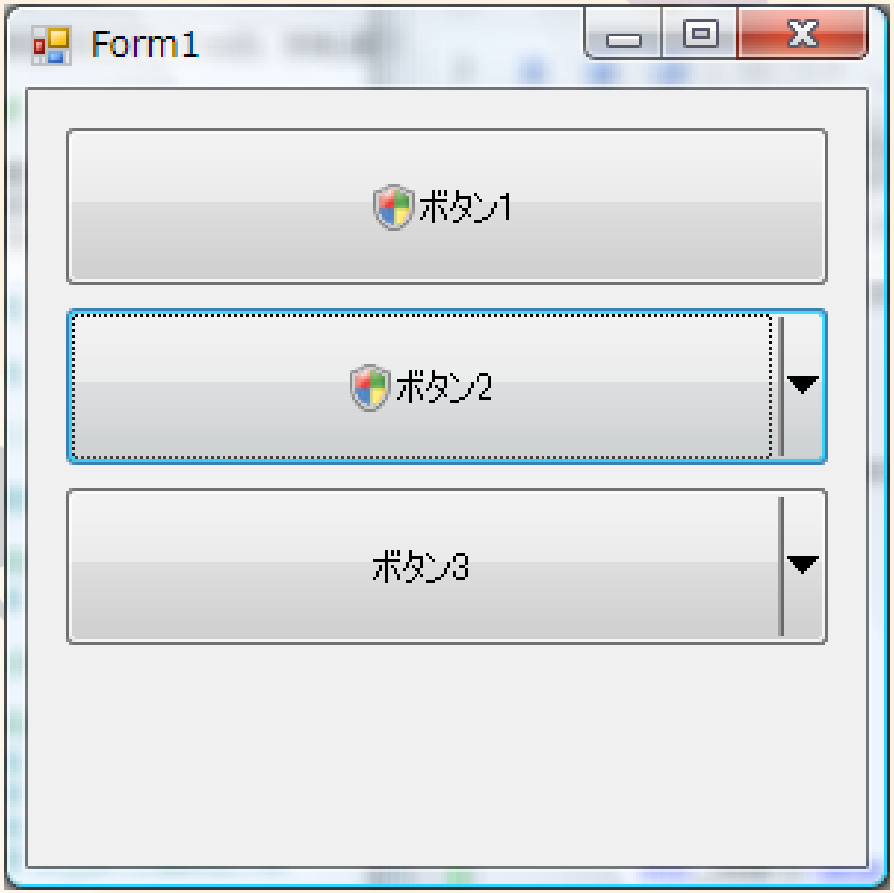

```
SendMessage(new HandleRef(this, this.Handle),
    BCM_SETSHIELD, IntPtr.Zero, true);
```

# 他には

const int
BS_SPLITBUTTON =
0x0000000C;

LiveSerachで検索すると、このボタンのサンプルがないって本家MSDNFにかかれているだけ。

**謎機能です。**



そして

# これらすべてを簡単に使える

# WankumaButton

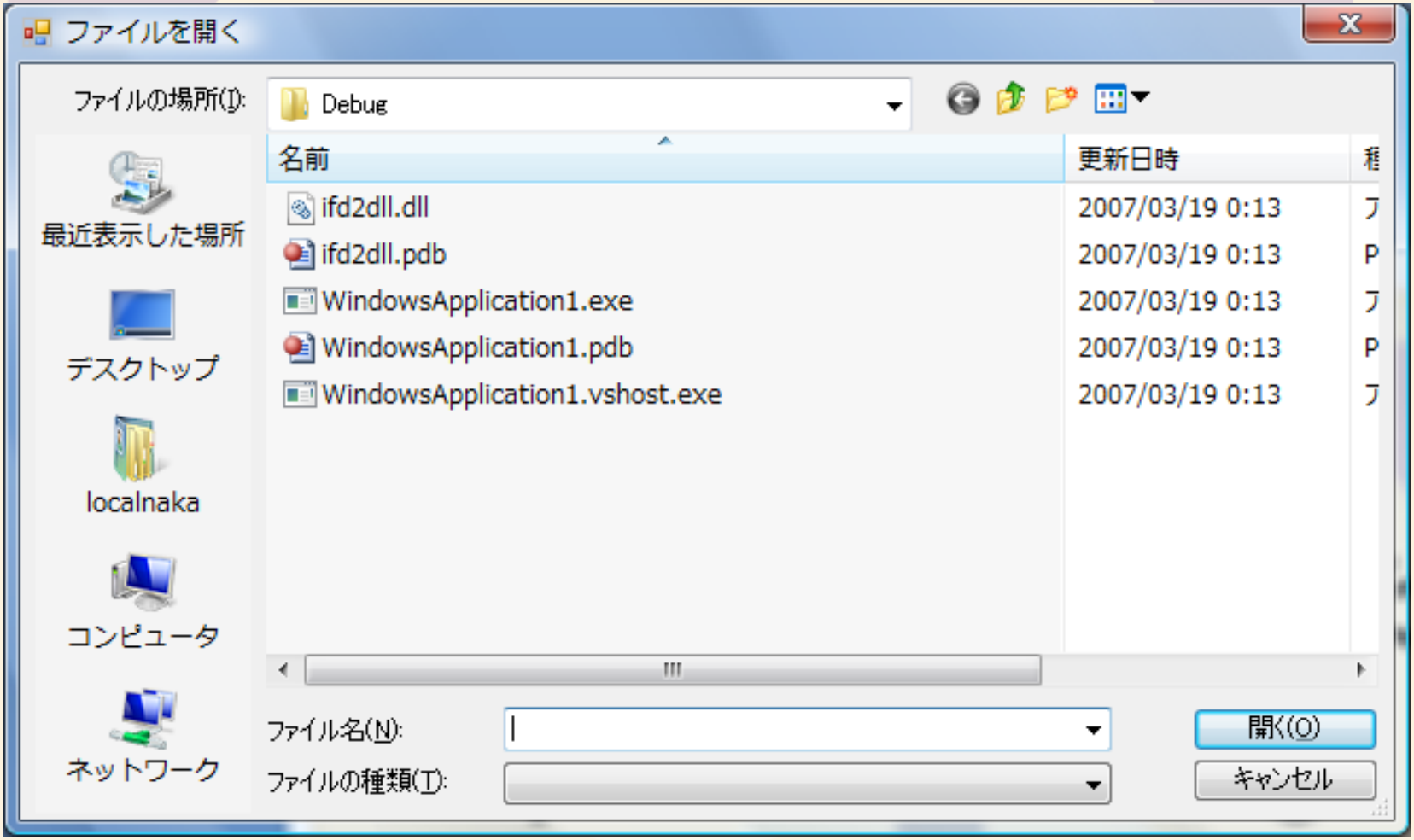
# Demo3

# 今回取り上げる機能は

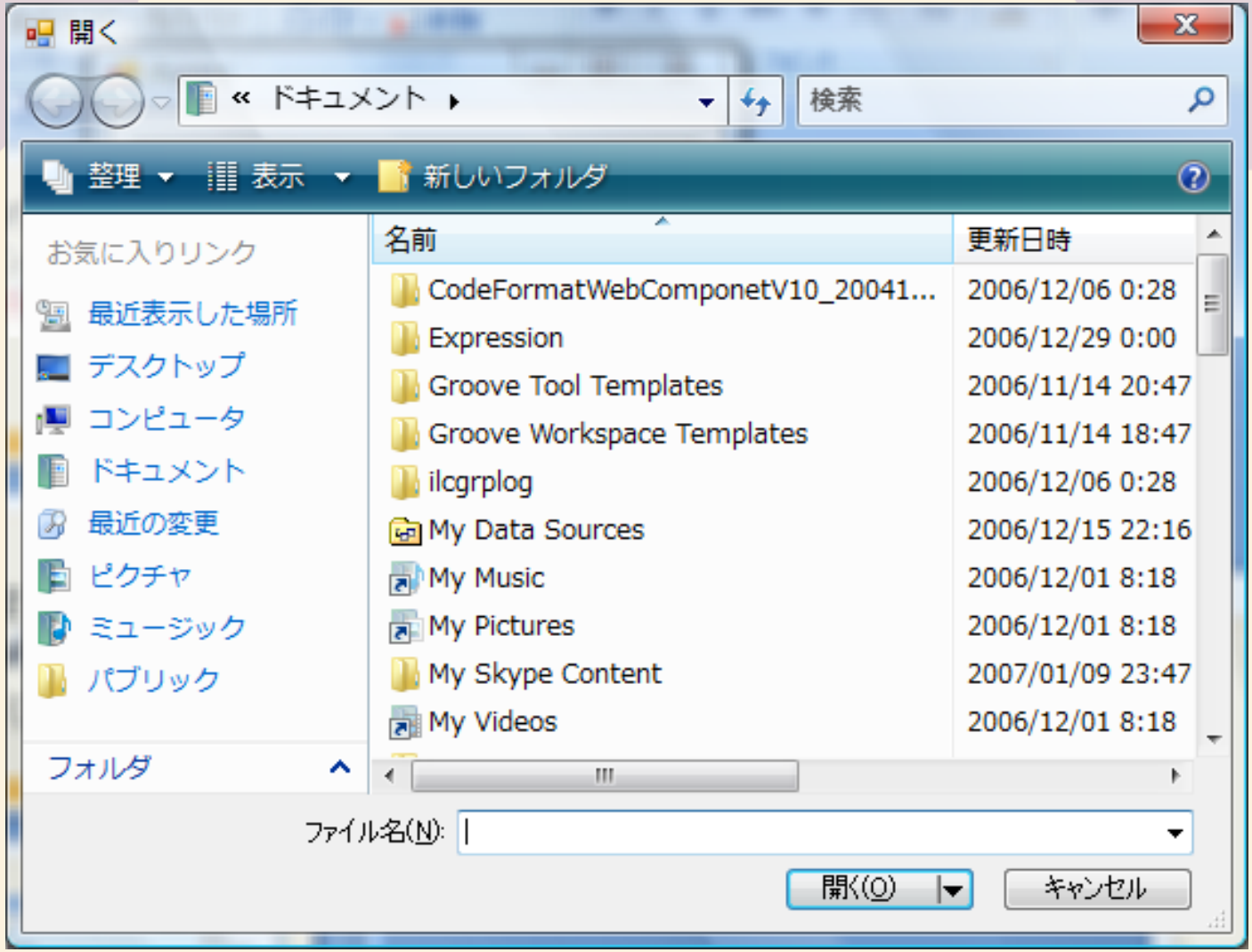
- ボタン
- IFileDialog
- System.IO.Log(CLFS)
- XPS

# いままで使ってきた OpenFileDialog はもう古い

# 古いの


ファイルを開く
ファイルの場所(I):
Debug
最近表示した場所
デスクトップ
localnaka
コンピュータ
ネットワーク
名前
更新日時
ifd2dll.dll
2007/03/19 0:13
ifd2dll.pdb
2007/03/19 0:13
WindowsApplication1.exe
2007/03/19 0:13
WindowsApplication1.pdb
2007/03/19 0:13
WindowsApplication1.vshost.exe
2007/03/19 0:13
ファイル名(N):
ファイルの種類(T):
開く(O)
キャンセル

# 新しいの


開く
« ドキュメント
検索
整理
表示
新しいフォルダ
お気に入りリンク
最近表示した場所
デスクトップ
コンピュータ
ドキュメント
最近の変更
ピクチャ
ミュージック
パブリック
フォルダ
名前
更新日時
CodeFormatWebComponetV10_20041... 2006/12/06 0:28
Expression 2006/12/29 0:00
Groove Tool Templates 2006/11/14 20:47
Groove Workspace Templates 2006/11/14 18:47
ilcgrplog 2006/12/06 0:28
My Data Sources 2006/12/15 22:16
My Music 2006/12/01 8:18
My Pictures 2006/12/01 8:18
My Skype Content 2007/01/09 23:47
My Videos 2006/12/01 8:18
ファイル名(N):
開く(O)
キャンセル

# .NET3.0ではサポートされてません。(XPでも動くしね3.0は)

# COMで実装されていますが、IDispatchはなく、IUnknownだけです。

CreateObject(“Shell32.FileDialog”)は無理ってこと

# C++/CLIの
# 出番じゃないか

# 使うためには

- Windows SDK 6.1をインストール
  http://www.microsoft.com/downloads/details.aspx?FamilyID=ff6467e6-5bba-4bf5-b562-9199be864d29&DisplayLang=en
- プロジェクトメニュー→プロパティ→構成プロパティ→C/C++→全般→追加のインクルードディレクトリにSDKのIncludeフォルダを追加
- #define設定
  - #define WINVER 0x0600
  - #define _WIN32_WINNT 0x0600

# ミニマムコード(ATL版)

```
::CoInitialize(NULL);
CComPtr<IFileOpenDialog> ifd;
ifd.CoCreateInstance(__uuidof(FileOpenDialog));
ifd->Show(NULL);
::CoUninitialize();
```

#もちろん開くだけ

# DEMO4
(ifd2)

# 今回取り上げる機能は

- ボタン
- IFileDialog
- System.IO.Log(CLFS)
- XPS

# ログを取る場合の問題点

- http://blogs.wankuma.com/naka/archive/2007/03/15/66950.aspx
- http://blogs.wankuma.com/naka/archive/2007/03/16/67146.aspx
- http://blogs.wankuma.com/naka/archive/2007/03/19/67563.aspx
- http://blogs.wankuma.com/naka/archive/2007/03/27/69201.aspx
- http://blogs.wankuma.com/naka/archive/2007/03/28/69340.aspx

こっそりあおり連載でした。

- 時系列に取れない(Mixされる)
- 遅い
- 大きい
- 排他待ち
- バラバラ

# CLFSをあなたは知っているか？

- Common Log File Systemという
- Windows Server 2003 R2から採用
- Windows SDKにいつまでも情報が載らず
- やっと使えるSDKが出たと思うと。
- .NET 3.0でこっそりとSystem.IO.Logとして実装されていた。
- 確かに当初からWinFXとして提供予定だったけど。

# CLFSとは

- 高速である
- 柔軟である
- 排他処理しなくてよい

利用するには

# System.IO.Log.Dllを参照設定するだけ。

# DEMO5
(clfs3)

# CLFSの使い方1

開く

```
using (LogRecordSequence sequence
    = new LogRecordSequence(
LOG_PATH,
System.IO.FileMode.OpenOrCreate,
System.IO.FileAccess.ReadWrite,
FileShare.ReadWrite))
```

ポイント

FileShare.ReadWriteで開く!!でないと排他がかかっちゃうよ

# CLFSの使い方2

初回だけポリシー設定を

```
if (sequence.LogStore.Extents.Count == 0) {
sequence.LogStore.Policy.AutoGrow = true;
sequence.LogStore.Policy.GrowthRate
      = new PolicyUnit(1, PolicyUnitType.Extents);
sequence.LogStore.Policy.Commit();
sequence.LogStore.Policy.Refresh();
```

ポイント

CommitとRefreshを実行する。

ほかにもファイルプレフィックス、サフィックスなどを設定する。

組み合わせで動かない場合もあるので、テストしましょう。

# CLFSの使い方3

初回エクステントの設定

```
sequence.LogStore.Extents.Add(EXTENT_NAME +
    sequence.LogStore.Extents.Count, EXTENT_SIZE);
```

ポイント

初回だけしかしない。

以後はポリシーのAutoGrowにまかせる。

最小サイズは512KB

でもそれなりのサイズにしましょう

# CLFSの使い方4

ログ出力

```
sequence.Append(
    segment,
    SequenceNumber.Invalid,
    SequenceNumber.Invalid,
    RecordAppendOptions.None);
```

ポイント

RecordAppendOptionsはNoneにすること。
でないとパフォーマンスはでない

## Vista以外の場合どうする？

LogRecordSequence
を
FileRecordSequence
にかえる。
複数書き込みできない
ポリシーベースの自動拡張できない
普通のファイルベース出力をしてくれる。

# 今回取り上げる機能は

- ボタン
- IFileDialog
- System.IO.Log(CLFS)
- XPS

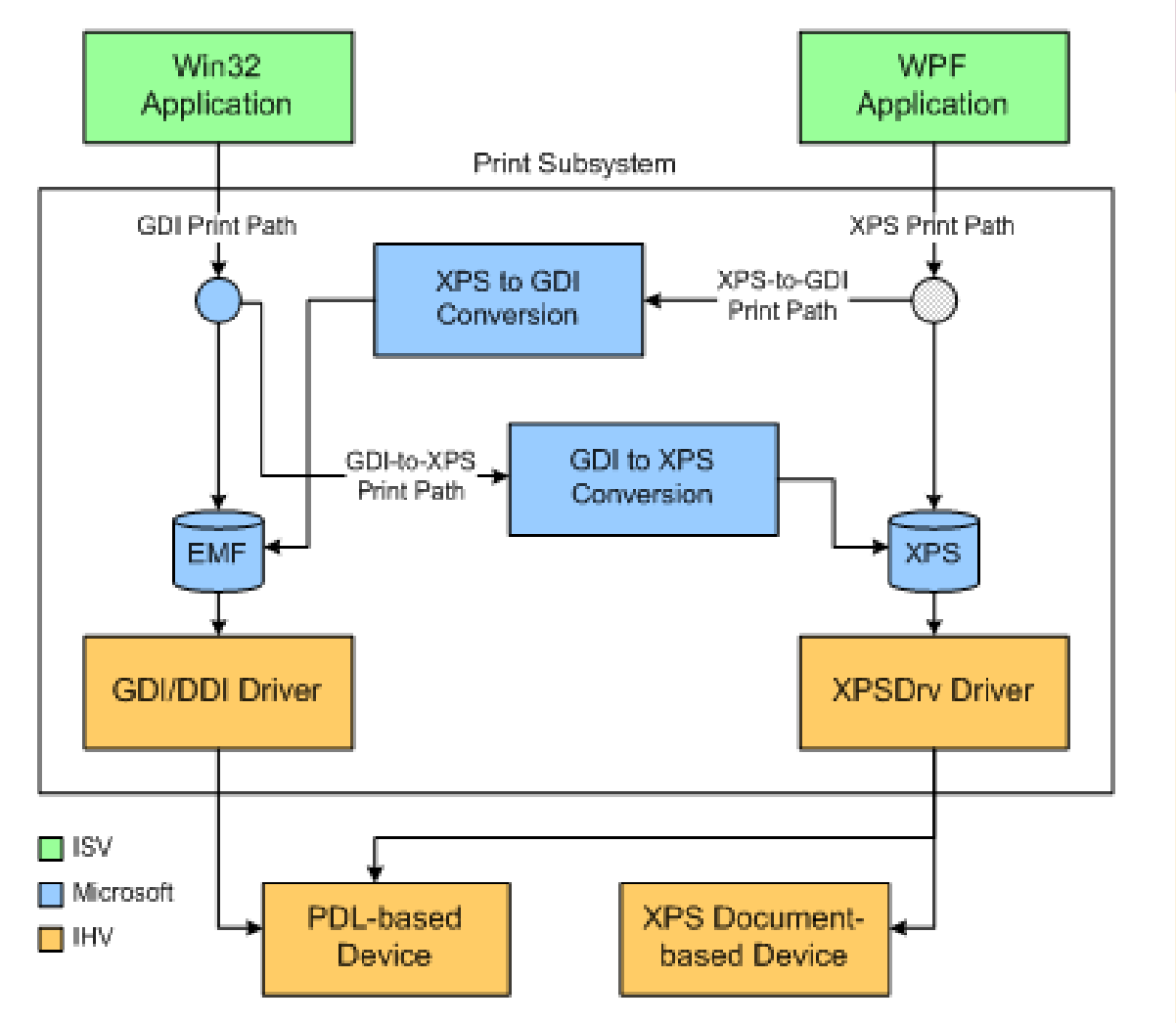
# XML Paper Specification

- PDFキラー
- Xamlのサブセットであり、内部は.docxと似た感じ

# XPSはどうやったら使えるの？

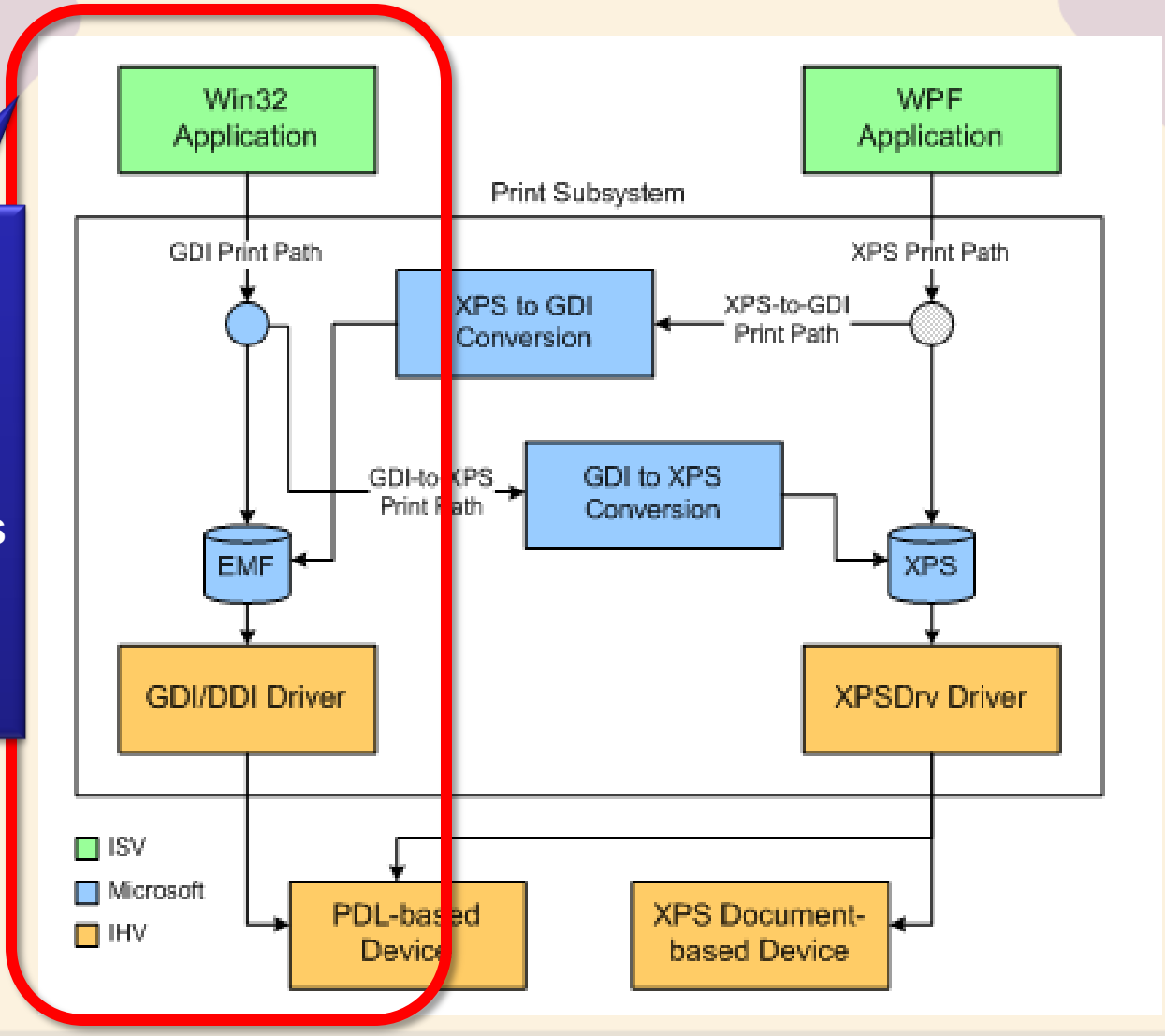
- 作る方
  - Windows Vista
  - 2007 Office Systems+保存アドイン
  - .NET Framework 3.0
- 見る方
  - Windows Vista
  - Windows XP or Windows Server 2003
    &
    XPS Essentials Pack + MSXML6.0

# XPSって従来の印刷とどう違うの？(Windows SDKより)

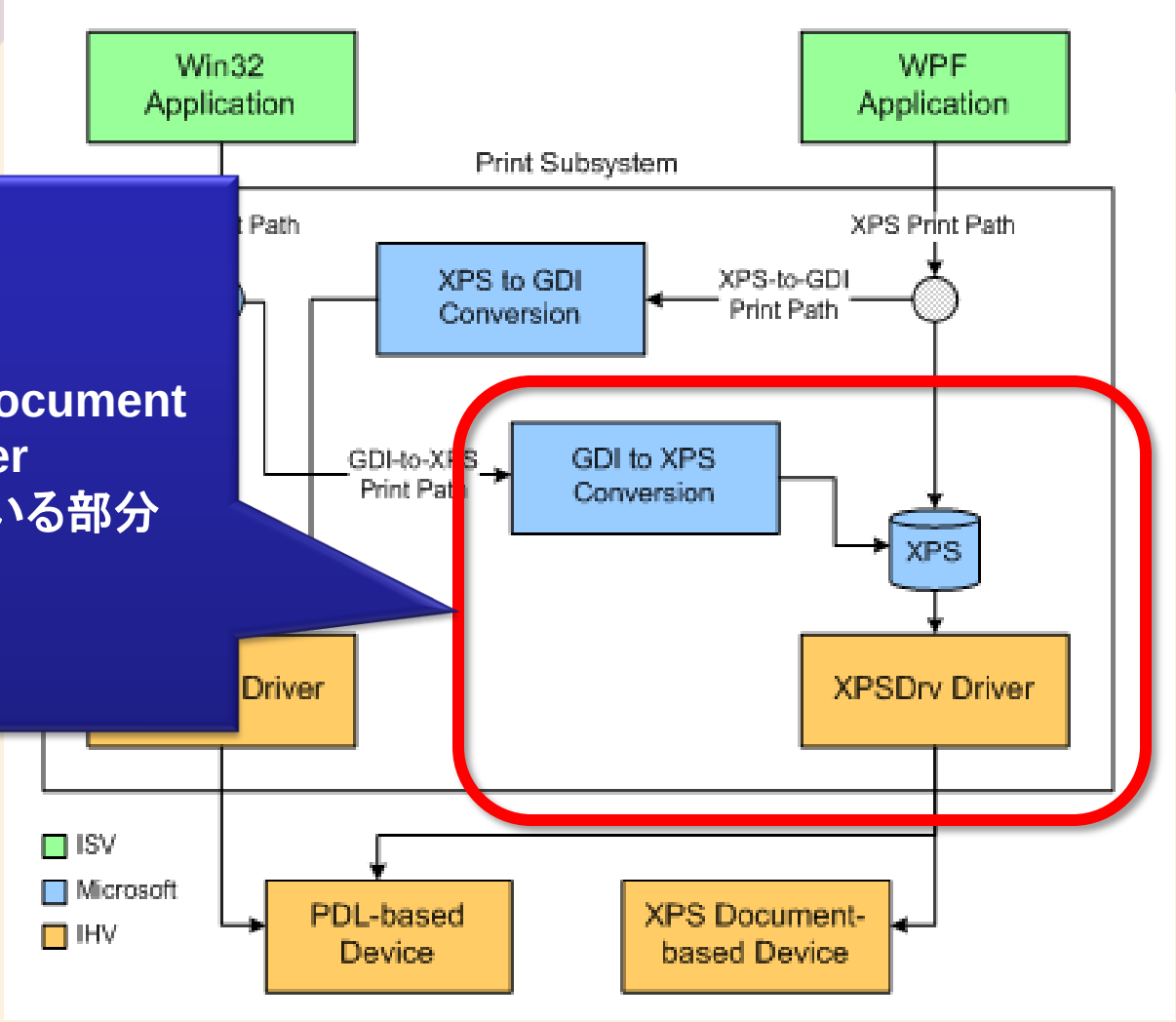


# XPSって従来の印刷とどう違うの？

**従来の Windows + GDIの印刷 (含Windows Forms)**

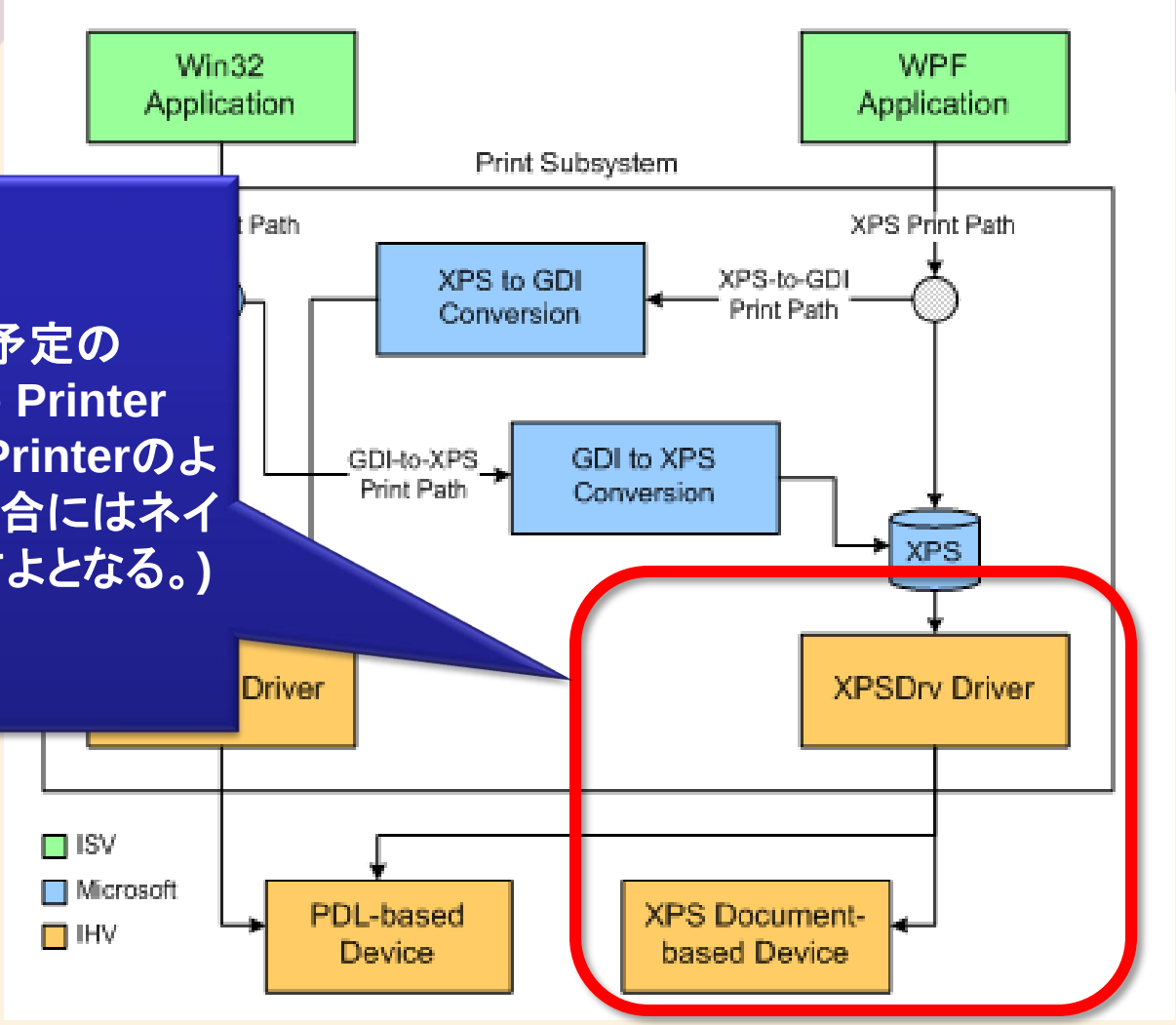


# XPSって従来の印刷とどう違うの？



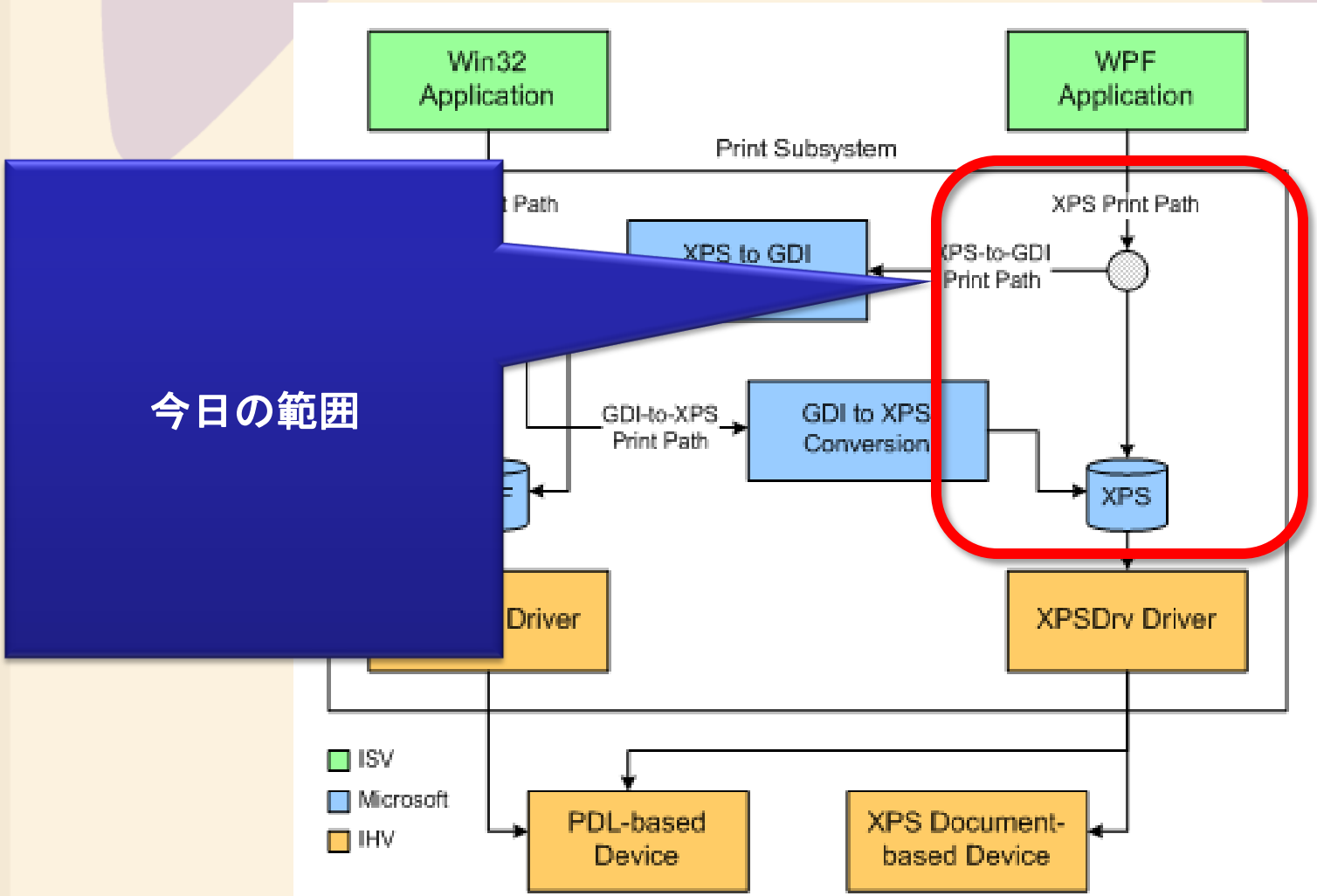

**Microsoft Document Writer の担当している部分**

# XPSって従来の印刷とどう違うの？




今後出る予定の
XPS Native Printer
(Post Script Printerのように、XPSの場合にはネイティブ対応ですよとなる。)

# XPSって従来の印刷とどう違うの？

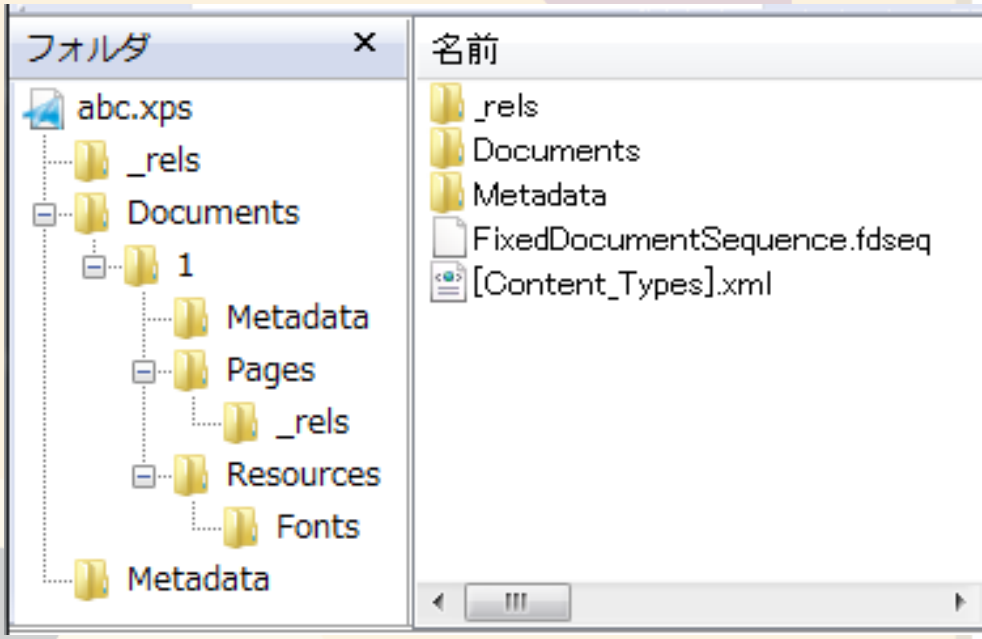


まずは出してみましょう

# DEMO6

# XPSってどうなっているの？

- ZIP圧縮されたXMLの集合体
- 各ページ情報は/Documents/1/Pages/1.fpageに格納
- リソースはフォントと、イメージ
- /Documents/1/Resourcesや、/Resourcesに格納



# ページ情報はどうなっているの？(抜粋)

- <FixedPage Width="793.76" Height="1122.56" xmlns="http://schemas.microsoft.com/xps/2005/" xml:lang="und">
- <Path Data="F1 M 75.52,75.52 L 165.28,75.52 165.28,86.08 75.52,86.08 z" Fill="#ffffffff" />
- <Glyphs Fill="#ff000000" FontUri="/Documents/1/Resources/Fonts/10-1D11-4DEE-9484-91F85C6F520A.odttf" FontRenderingEmSize="10.5604" StyleSimulations="None" OriginX="75.52" OriginY="84.64" Indices="2102;2104;2106;2108;2110;59;51;54;11764;11754" UnicodeString="あいうえおXPS薔薇" />
- </FixedPage>

ページ情報

ブラシ

文字

フォント指定

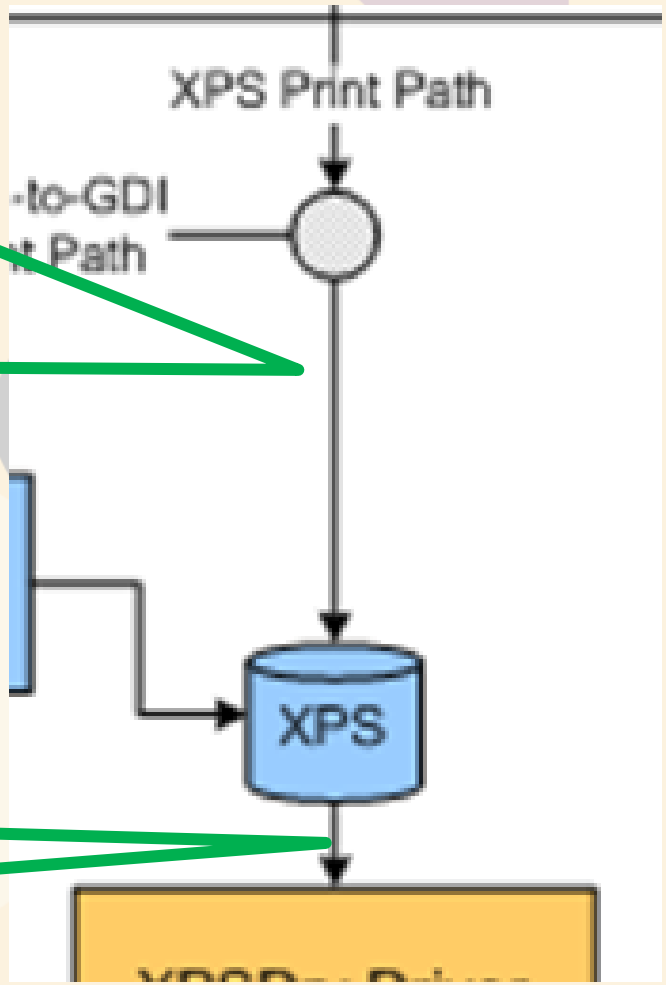
サイズ、位置

文字列

# それではXPSを作ってみましょう。

関連する名前空間は

- System.Windows
  - .Xps
  - .Xps.Packaging
  - .Xps.Serialization
- System.Printing



# それではXPSを作ってみましょう。

XPSを作ってみましょう。

WPF関係の参照設定がされている前提

System.Printing.dll

ReachFramework.dll

の2つがコアになるコンポーネント

作ってみましょう

# DEMO7

(xps1)

# XPSを作るポイント

- 日本語を扱うならフォントサブセットは必須！！
- 縦書きは頼るな。(使えない)
- 仕様書をよく読む。
  - 結局出力するのはXML

Microsoft

# ready
# for a new day

Microsoft Office

Windows Vista

Microsoft Exchange Server 2007

# 参考文献

**★CommandLinkとSETNOTEとシールドアイコン**

**knom's developer corner(en)**

http://blogs.msdn.com/knom/archive/2007/03/12/command_5F00_link.aspx

**WindowsSDKの位置**

Win32 and COM Development→User Interface →Windows Controls →Individual Control Information →Button Controls

# 参考文献

**★IFileDialogについて**

**Windows Vista for Developers – Part 6 – The New File Dialogs(en)**

http://weblogs.asp.net/kennykerr/archive/2006/11/10/Windows-Vista-for-Developers-_1320_-Part-6-_1320_-The-New-File-Dialogs.aspx

Kkamegawaさんのcodeseekでの発表資料(ja)

http://mist.clueup.org/files/default.aspx

**WindowsSDKの位置**

Win32 and COM Development→User Interface →Windows Shell→Shell Reference→Shell Interfaces →IFileDialog

# 参考文献

**★System.IO.Log(CLFS)について**

**Fast and Flexible Logging with Vista's Common Log File System(en)**

http://www.devx.com/VistaSpecialReport/Article/33848/0/page/1

**Windows Server 2003 R2 の新機能**

http://technet2.microsoft.com/WindowsServer/ja/Library/f9d70026-ae8b-4969-8755-1ea1edc4e38e1041.mspx?mfr=true

**Windows SDKの位置**

Win32 and COM Development →System Services→File Systems→Common Log File System

.NET Framework Development→.NET Framework Technologies→Core Development  Technologies→Logging Support in System.IO.Log

# 参考文献


**★XPSについて1**

**Printing Overview**

ms-help://MS.MSSDK.1033/MS.NETFX30SDK.1033/wpf_conceptual/html/0de8ac41-9aa6-413d-a121-7aa6f41539b1.htm

**2007 Microsoft Office プログラム用 Microsoft PDF/XPS 保存アドイン**

http://www.microsoft.com/downloads/details.aspx?FamilyID=4d951911-3e7e-4ae6-b059-a2e79ed87041&DisplayLang=ja

**Microsoft XML Paper Specification Essentials Pack Version 1.0**

http://www.microsoft.com/downloads/details.aspx?FamilyId=B8DCFFDD-E3A5-44CC-8021-7649FD37FFEE&displaylang=en

**Microsoft Core XML Services (MSXML) 6.0**

http://www.microsoft.com/downloads/details.aspx?displaylang=ja&FamilyID=993c0bcf-3bcf-4009-be21-27e85e1857b1

# 参考文献


**★XPSについて2**

**Windows HardwareDeveloper Central**

http://www.microsoft.com/whdc/xps/default.mspx

**XPS for Application Developers**

http://www.microsoft.com/whdc/xps/xpsappdevs.mspx

**XML Paper Specification (Spec)**

http://www.microsoft.com/whdc/xps/xpsspec.mspx

**XPS Team Blog**

http://blogs.msdn.com/xps/

**Feng Yuan (袁峰)**

http://blogs.msdn.com/fyuan/

# 参考文献


**★XPSについて3**

**Optimize XPS markup(パフォーマンスの注意点)**

http://blogs.msdn.com/fyuan/archive/2006/01/18/514450.aspx

**Query regarding CreateFontPackage API used for Font Subsetting**

http://forums.microsoft.com/MSDN/ShowPost.aspx?PostID=222335&SiteID=1

**Insertion of an adendum to the intro to font embedding**

http://blogs.msdn.com/michkap/archive/2006/08/02/686538.aspx

Creating an XPS Document Sample

ms-help://MS.MSSDK.1033/MS.NETFX30SDK.1033/wpf_samples/html/a7736471-4322-40ea-8580-34c8eb0dbb3e.htm