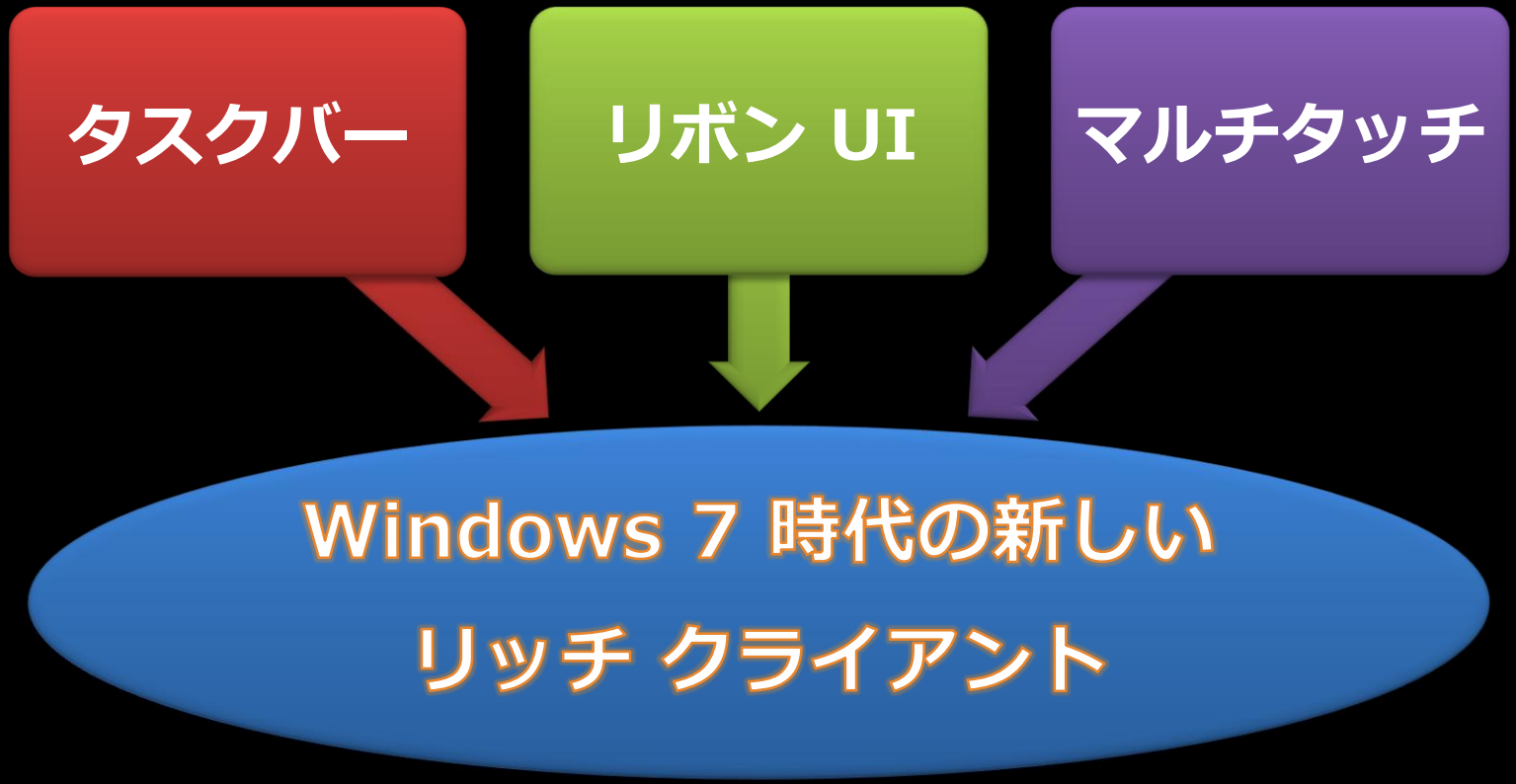
Future
Technology Days

Windows 7

追加された新たな API

マイクロソフト株式会社
デベロッパー & プラットフォーム 統括本部
井上 大輔、井上 章

What's New ?
Windows 7 APIs



Windows XP / Vista にはないユーザー体験 (UX) を！！

Future
Technology Days

Windows 7 で追加された新たな API

マルチタッチ 編

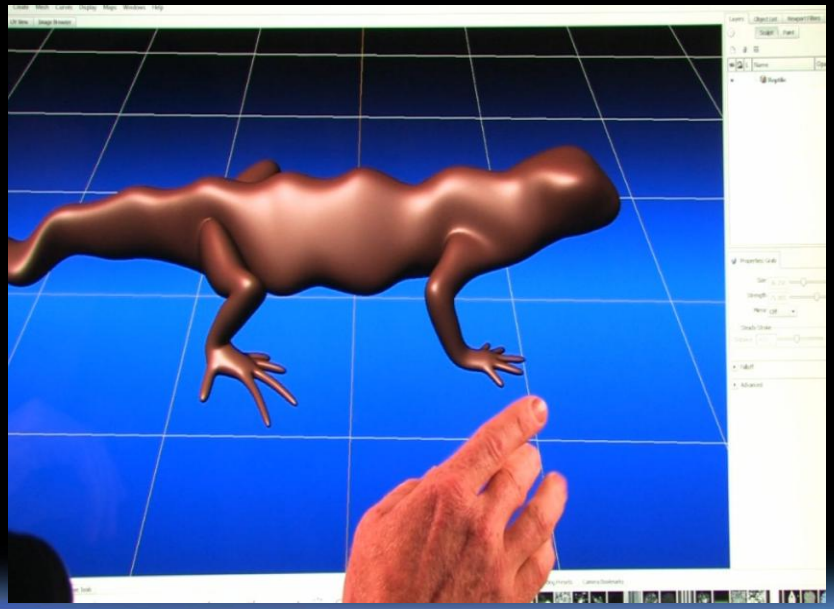
マイクロソフト株式会社
デベロッパー & プラットフォーム 統括本部
エバンジェリスト
井上 大輔（いのうえ だいすけ）

Agenda

- マルチタッチで出来ること
- マルチタッチ環境
- マルチタッチ API

Autodesk: Mudbox And AutoCAD 2009

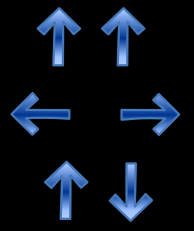
Partner Video



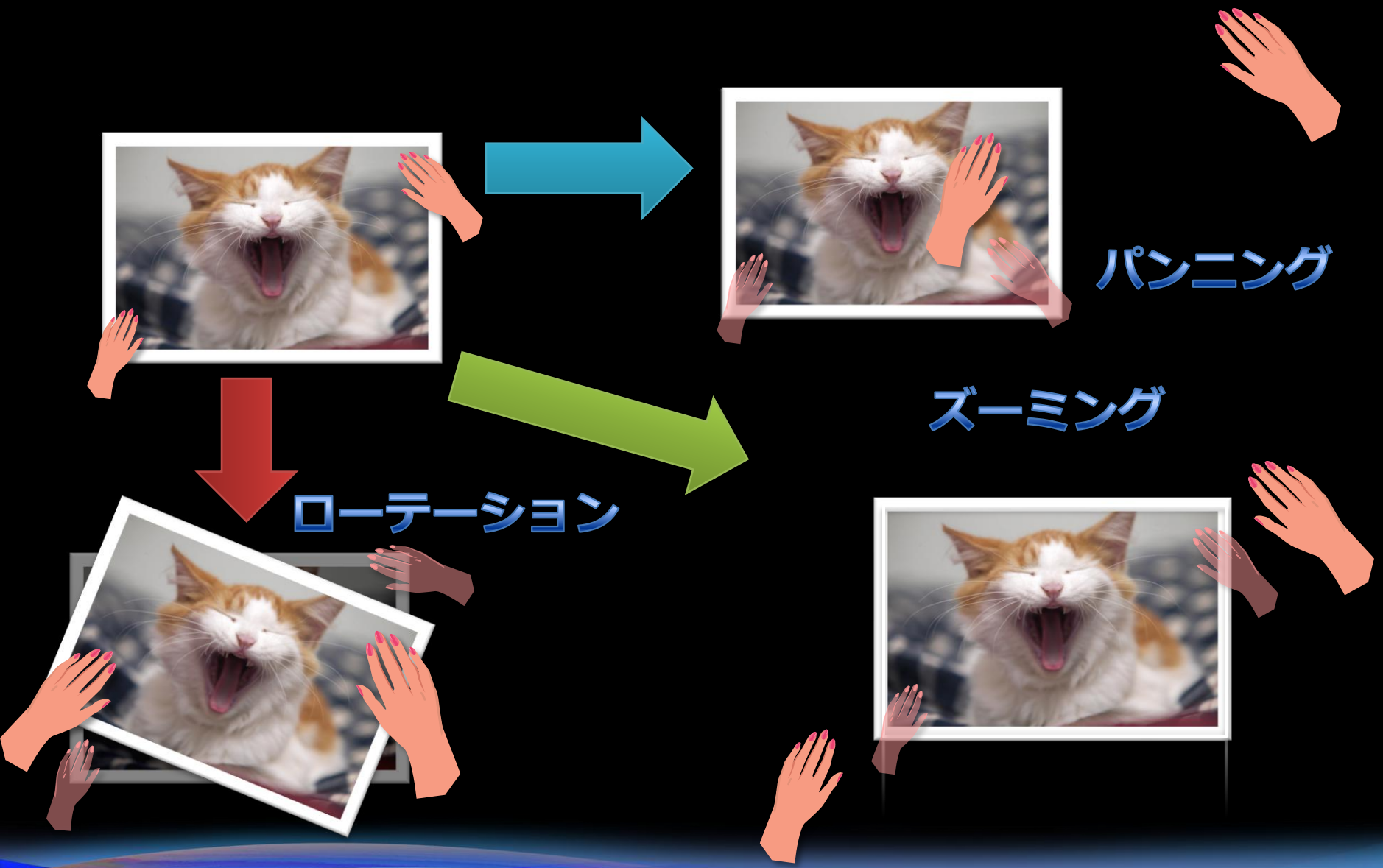

partner video
Autodesk
Windows 7
And Multi-Touch

マルチタッチで出来ること

- **タッチ**であること
  - フィンガータッチオペレーション
  - マウスと同じ動作（1次元の入力）
    - ドラッグ：スクロール、移動
- **マルチ**であること
  - 複数の同時ポインタ
  - マウスでできなかった新しい動作
    - パン：平行移動
    - ズーム：拡大縮小
    - ローテーション：回転



マルチタッチ操作



Windows 7 : New APIs

demo

マルチタッチで出来ること

マルチタッチ環境

マルチタッチ環境

ハードウェア

- マルチタッチ対応のハードウェアはすでにいくつか市場で発売済み
- 今後対応デバイスは拡大予定

HP TouchSmart 2

ソフトウェア

- Windows 7
- Win32 API
- .NET 4.0

Dell Latitude XT

メリット

- 新たなPCの利用方法の提供
- 新しいユーザーエクスペリエンス
- 新しい価値の提供

マルチタッチAPI

マルチタッチAPI

- **タッチ**であること
  - **WM_TOUCH*** 関連メッセージ
    - タッチ操作に最適化したエクスペリエンスを実現
      - WM_TOUCHDOWN
      - WM_TOUCHMOVE
      - WM_TOUCHUP
- **マルチ**であること
  - **WM_GESTURECOMMAND**
    - 合成されたジェスチャーコマンド

Windows 7 : New APIs

demo

マルチタッチ アプリケーション開発

WM_GESTURECOMMAND

送信される情報:

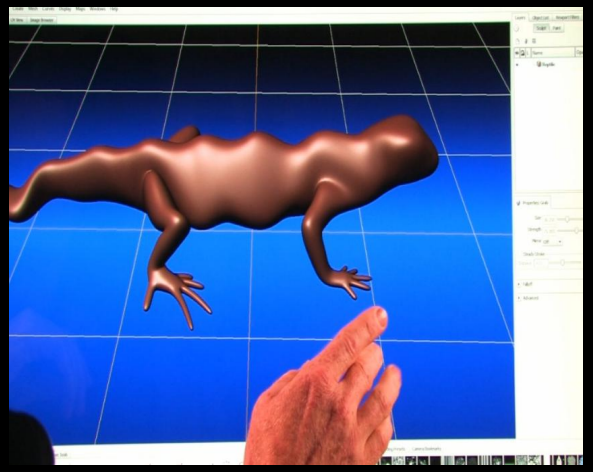
Command	Location	Argument
パン	中心点	移動距離
ズーム	中心点	移動距離
ローテート	中心点	絶対角（開始時） 回転角（更新時）
2点タップ	中心点	NA
右クリック ジェスチャー	最初のタッチ点	NA

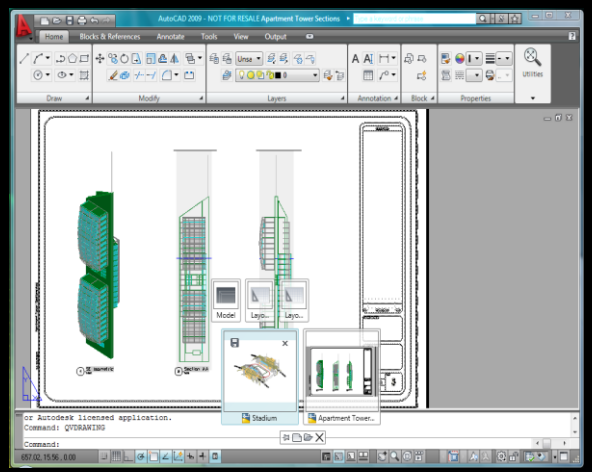
プラットフォーム毎の開発方法

	標準機能	GESTURE	TOUCH
Win32	• 標準のスクロールバーでの操作	• **WM_GESTURE メッセージ**	• **WM_TOUCH メッセージ** • COM ベースの操作,慣性プロセッサ
WPF	• WPF 4.0 ScrollViewerでのスクロール操作をサポート	• **Gesture イベント** • 慣性の設定	• **Touch イベント** • 操作,慣性プロセッサ
WinForms	• 標準のスクロールバーでの操作	• WM_GESTURE メッセージ • P/Invoke	• Inkコントロール対応操作、慣性プロセッサ • リアルタイム スタイラス、インクコレクター

マルチタッチの時代がやってくる

- タッチは直観的で使い慣れたインタフェース
- コンピューターに不慣れな人でも簡単操作
- 業務系アプリケーションにも効果的
- ISVs には好反応
- すでに対応を開始しているISV:
  - AutoDesk
  - Avoco Secure
  - Corel
  - Cyberlink
  - Fuel Industries
  - Identity Mine
  - Sonic





Future
Technology Days

Windows 7 で追加された新たな API

タスクバー & リボン UI 編

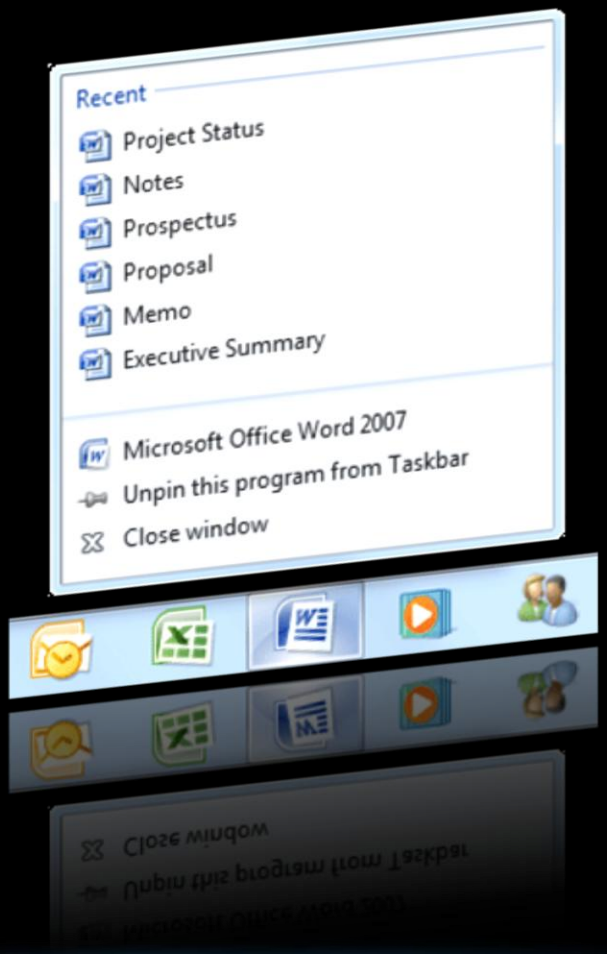
マイクロソフト株式会社
デベロッパー & プラットフォーム 統括本部
エバンジェリスト
井上 章（いのうえ あきら）

タスクバー

新しいタスクバー

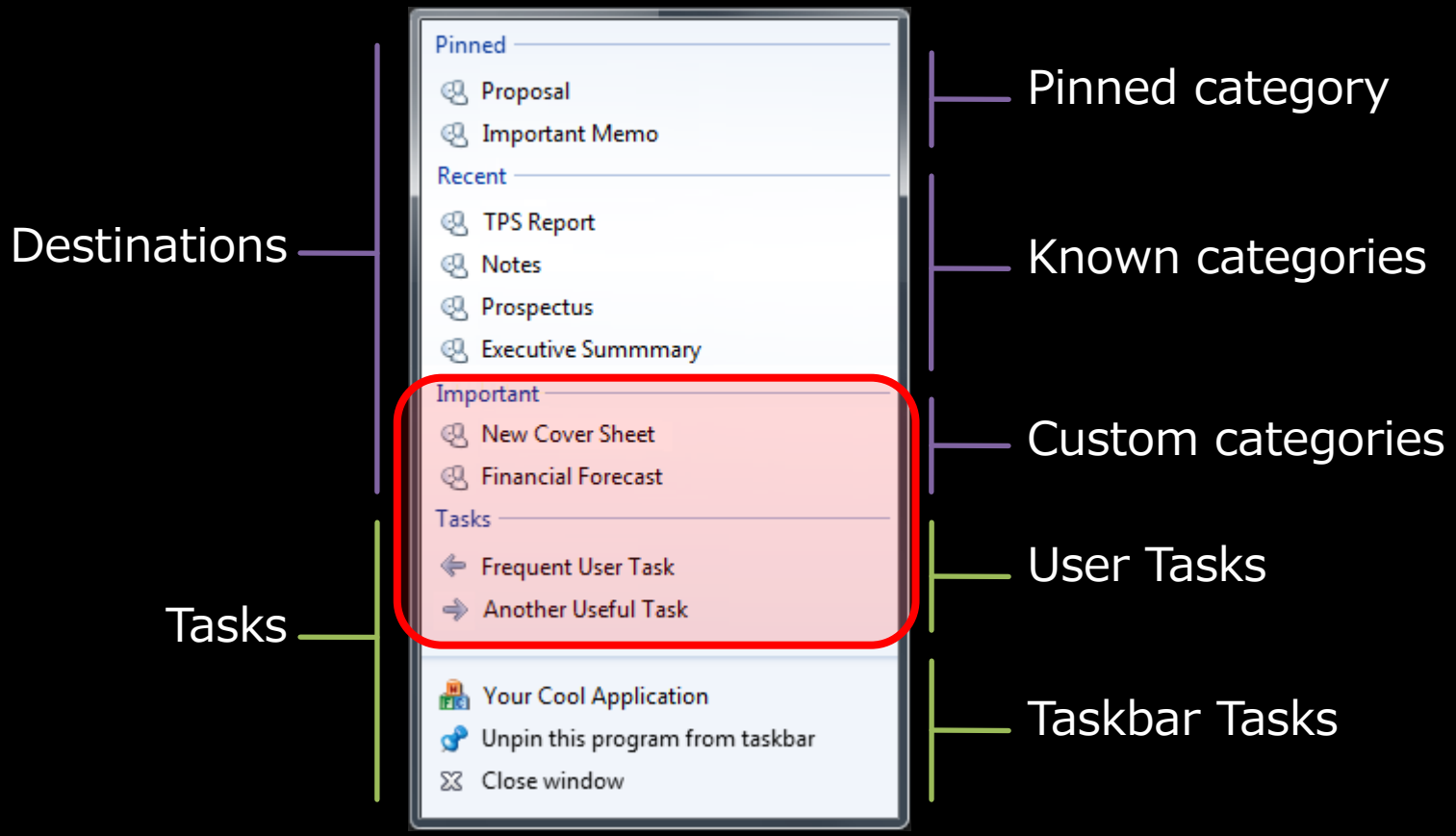
- 新しい実行ポイント：ジャンプリスト
- コマンドを可視化したサムネールツールバー
- ステータス表示の新しいやり方
  オーバーレイ アイコン・プログレスバー
- カスタム スイッチャーの表示
- Application Identity (AppID) による管理
- デスクバンド、ウィンドウ配置、ガジェット
- 今後のプラットフォームサポート

ジャンプリスト
概要



- プログラムのための小さな Start Menu
- 操作対象のファイルやタスク等をリスト表示
- カスタマイズ可能
- ドラッグや右クリックからアクセス可能
- API から見た名前は “Destination List”

ジャンプリスト


Pinned
Proposal
Important Memo
Recent
TPS Report
Notes
Prospectus
Executive Summmary
Important
New Cover Sheet
Financial Forecast
Tasks
Frequent User Task
Another Useful Task
Your Cool Application
Unpin this program from taskbar
Close window
Destinations
Tasks
Pinned category
Known categories
Custom categories
User Tasks
Taskbar Tasks

ジャンプリスト
カスタマイズ

- コンテンツはカスタマイズ可能
- Destinations（対象欄）
  - **Custom categories**（自分で提供可能）
  - Known categories（最近使ったファイル）
- Tasks（操作欄）
  - **User Tasks** (自分で提供可能)
  - Tasks は自由に Pin できるわけではない
- API（通常パターン）
  - User Tasks ： IShellLinks
  - Destinations ： IShellItems

ジャンプリストの追加

```
ICustomDestinationList cdl
 = (ICustomDestinationList)new CDestinationList();

uint maxslots;
object oa;

cdl.BeginList(
  out maxslots,
  ref DesktopIntegration.IID_IObjectArray,
  out oa);

IObjectCollection oc
 = (IObjectCollection)new CEnumerableObjectCollection();
oc.AddObject(
  DesktopIntegration.CreateUserTask("Task 1", "/task1"));

cdl.AddUserTasks(oc);
cdl.CommitList();
```

ジャンプリストの追加 (XAML)

```
<Application x:Class="Win7Samples.App"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:win7="clr-namespace:Microsoft.Win7.Bridge;assembly=Win7Bridge"
    StartupUri="Window1.xaml">

  <win7:JumpList.JumpList>
    <win7:JumpList ShowFrequentCategory="True" ShowRecentCategory="False">
      <win7:JumpTask
        ApplicationPath="NotePad.exe"
        IconResourcePath="NotePad.exe" IconResourceIndex="0" Arguments=""
        Title="メモ帳" Description="メモ帳を起動する" />

      <win7:JumpPath
        CustomCategory="オリジナルカテゴリ" Path=".¥Readme.txt"
        Title="Read Me File" Description="ReadMe テキストを開く" />
    </win7:JumpList>
  </win7:JumpList.JumpList>

</Application>
```

Windows 7 : New APIs

demo

ジャンプリスト

サムネール ツールバー
概要

- アプリケーション用の“リモコン“
- アプリケーションのコマンドとして機能
- 最大 7 つのボタンが利用可能
- タスクバーサムネールからアクセス可能

サムネール ツールバー
API とその利用方法

- **ITaskbarList3** インターフェースが追加
  - “TaskbarButtonCreated” メッセージを受ける
- 7 つのボタンのコレクションで構成され、それぞれは **THUMBBUTTON 構造体** で定義

```
typedef struct tagTHUMBBUTTON
{
  DWORD dwMask;
  UINT iId;          // ユニークなボタン ID
  UINT iBitmap;      // Imagelist ID
  HICON hIcon;       // ...もしくは HICON
  WCHAR szTip[260]; // Tooltip string
  DWORD dwFlags;     // Enable/disable/hide/etc
} THUMBBUTTON;
```

サムネールツールバー作成

```
if (msg == DesktopIntegration.WM_TaskbarButtonCreated)
{
    _pTaskbarList3 = (ITaskbarList3)new CTaskbarList();

    THUMBBUTTON[] buttons = new THUMBBUTTON[2];

    buttons[0].dwMask = THBMASK.THB_ICON | THBMASK.THB_TOOLTIP;
    buttons[0].iId = 100;
    buttons[0].hIcon = DesktopIntegration.LoadSystemIcon(
                                        SystemIcon.IDI_QUESTION);
    buttons[0].szTip = "Button 1";

    _pTaskbarList3.ThumbBarAddButtons(hwnd, 2, buttons);
}
```

サムネールツールバー作成 (XAML)

```
<Window x:Class="Win7Samples.Window1"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:win7="clr-namespace:Microsoft.Win7.Bridge;assembly=Win7Bridge">

   <win7:TaskbarItemInfo.TaskbarItemInfo>
      <win7:TaskbarItemInfo>
         <win7:TaskbarItemInfo.ThumbButtons>
            <win7:ThumbButtonInfoCollection>

    <win7:ThumbButtonInfo
     DismissOnClick="False" ImageSource="cut.png" Command="Cut"
     CommandTarget="{Binding ElementName=_textBox}"
     Description="{Binding RelativeSource={RelativeSource Self}, Path=Command.Text}"/>

            </win7:ThumbButtonInfoCollection>
         </win7:TaskbarItemInfo.ThumbButtons>
      </win7:TaskbarItemInfo>
   </win7:TaskbarItemInfo.TaskbarItemInfo>

</Window>
```

Windows 7 : New APIs

demo

サムネール ツールバー

タスクバー アイコン
概要

- 大/小アイコン表示
- 任意プログラムを常に表示可能 (Pin)
- Hot-track 表示

- オーバーレイ アイコン
  - プログラムアイコンに重ねてアイコンを表示

- プログレスバー
  - タスクバーアイコンの背景がプログレスバー表示

オーバーレイ アイコン
API と利用方法

- タスクバーボタンにオーバーレイアイコンを表示するには **ITaskbarList3** インターフェースを使用

```
HRESULT SetOverlayIcon(
    HWND hwnd,     // 対象ウィンドウ
    HICON hIcon,   // 表示するアイコン（NULL は非表示）
    LPCWSTR pszDescription);   // テキスト情報
```

オーバーレイアイコンの設定

```
private void button2_Click(object sender, RoutedEventArgs e)
{
    _pTaskbarList3.SetOverlayIcon(
        _hwnd,
        DesktopIntegration.LoadSystemIcon(
                            SystemIcon.IDI_ERROR),"ERROR");
}

private void button4_Click(object sender, RoutedEventArgs e)
{
    _pTaskbarList3.SetOverlayIcon(_hwnd, IntPtr.Zero, "");
}
```

オーバーレイアイコンの設定 (XAML)

```
<Window x:Class="Win7Samples.Window1"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:win7="clr-namespace:Microsoft.Win7.Bridge;assembly=Win7Bridge">

   <win7:TaskbarItemInfo.TaskbarItemInfo>

     <win7:TaskbarItemInfo
       ProgressState="{Binding ElementName=_progressState, Path=SelectedItem}"
       ProgressValue="{Binding ElementName=_progressSlider, Path=Value}"
       Description="{Binding ElementName=_textBox, Path=Text}"
       Overlay="{Binding ElementName=_overlaySelection, Path=SelectedItem.Source}">

     </win7:TaskbarItemInfo>

   </win7:TaskbarItemInfo.TaskbarItemInfo>

</Window>
```

Windows 7 : New APIs

demo

オーバーレイ アイコン

プログレスバー
API と利用方法

- タスクバーボタンにプログレスバーを表示するには **ITaskbarList3** インターフェースを使用

```
HRESULT SetProgressState(HWND hwnd,
                         TBPFLAG tbpFlag);
HRESULT SetProgressValue(
    HWND hwnd,
    ULONGLONG ullCompleted, // 進捗値
    ULONGLONG ullTotal      // 目標値 );
```

- CLSID_ProgressDialog からタスクバーの進捗値を取得できる

プログレスバーの設定

```
// プログレスバーの値
_pTaskbarList3.SetProgressValue(
      _hwnd,
      (uint)e.NewValue,
      (uint)slider1.Maximum);

// プログレスバーの状態
_pTaskbarList3.SetProgressState(
     _hwnd,
     TBPF_ERROR);
```

プログレスバーの設定 (XAML)

```
<Window x:Class="Win7Samples.Window1"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   xmlns:win7="clr-namespace:Microsoft.Win7.Bridge;assembly=Win7Bridge">

   <win7:TaskbarItemInfo.TaskbarItemInfo>

     <win7:TaskbarItemInfo
      ProgressState="{Binding ElementName=_progressState, Path=SelectedItem}"
      ProgressValue="{Binding ElementName=_progressSlider, Path=Value}"
      Description="{Binding ElementName=_textBox, Path=Text}"
      Overlay="{Binding ElementName=_overlaySelection, Path=SelectedItem.Source}">

     </win7:TaskbarItemInfo>

   </win7:TaskbarItemInfo.TaskbarItemInfo>

</Window>
```

Windows 7 : New APIs

demo

プログレスバー

カスタム スイッチャー
概要

- カスタム UI を表示 (TDI / MDI)
- 各ウィンドウ毎にカスタムサムネールを表示
- プログラムのウィンドウリストに現れる

カスタムスイッチャーの表示
TDI / MDI 子ウィンドウ

- スイッチャーとしてウィンドウを追加登録するために ITaskbarList3 を使う

```
HRESULT RegisterTab(HWND hwndTab,
                    HWND hwndFrame);
HRESULT UnregisterTab(HWND hwndTab);
HRESULT SetTabOrder(HWND hwndTab,
                    HWND hwndInsertAfter);
HRESULT SetTabActive(HWND hwndTab,
                     HWND hwndFrame,
                     TBATFLAG tbatFlags);
```

Windows 7 : New APIs

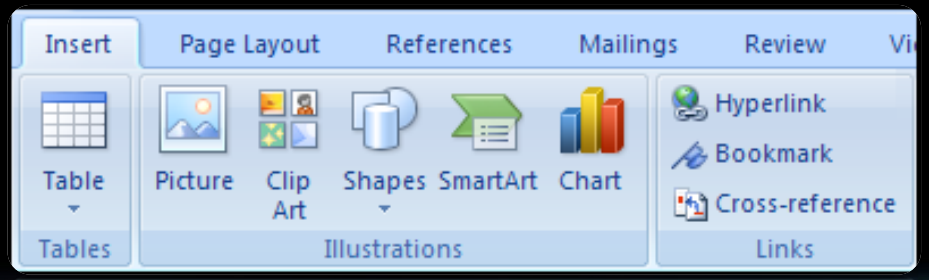
demo

カスタムスイッチャー

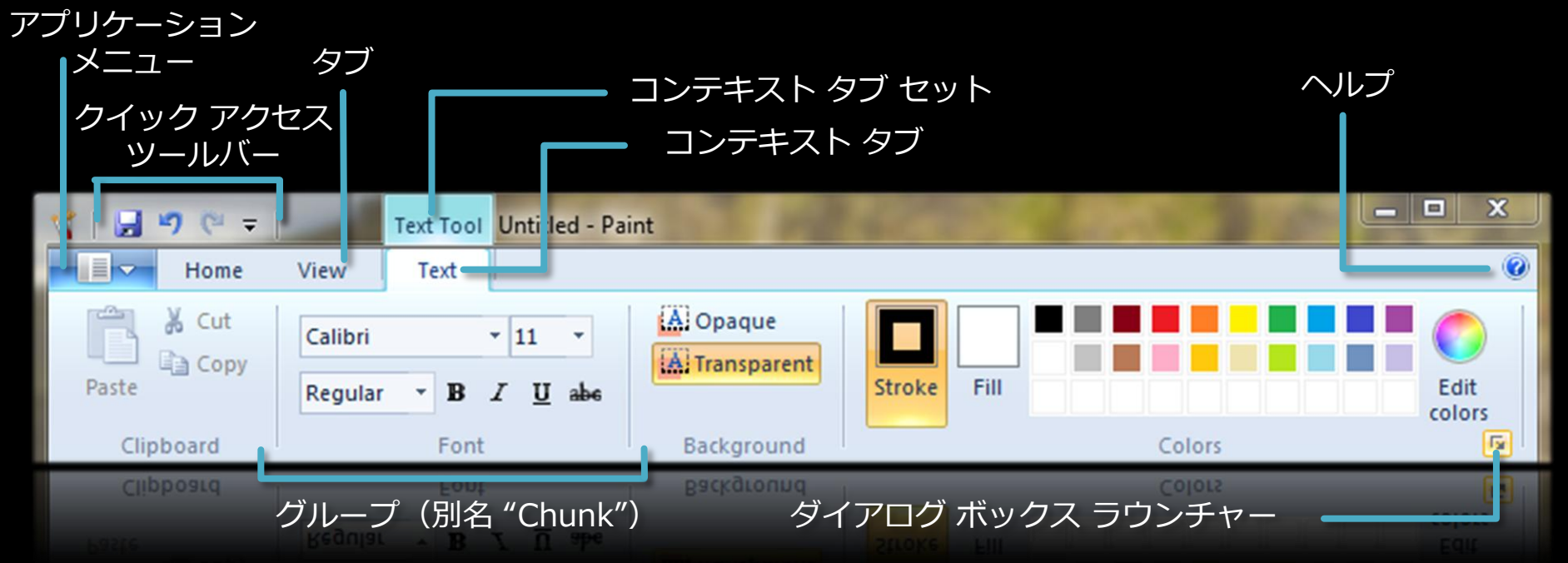
リボン UI

リボン UI とは？

- Office 2007 で採用された新しいユーザーインターフェース
- タブと大きめに配置されたコントロールで構成
- 従来のメニュー バーに置き換わる UI
- Windows 7 ではリボン UI のための新たな API が標準で用意され、アプリケーションで容易に実装できる
- Windows 7 の ペイントやワードパッドでリボン UI が採用
- 以下の 2 種類に分けられる
  - Office 2007 リボン
  - Windows (Scenic) リボン



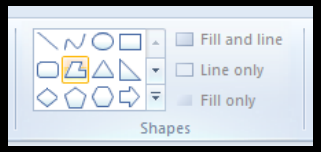
Windows (Scenic) リボン



- Windows 7 に標準搭載, Windows Vista に配置可能
- Win32 API, COM ベースの開発（まずはネイティブデベロッパを対象）
- Microsoft Office 2007 リボンとほぼ同等の機能を提供

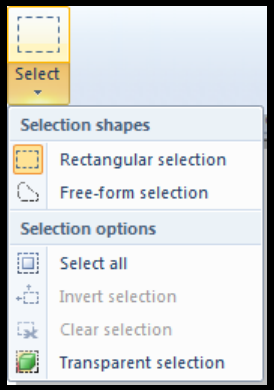
リボン コントロール

“In-Ribbon” ギャラリー

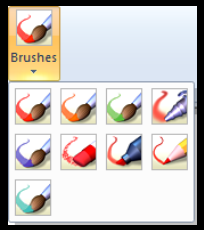


グループ ダイアログ ランチャー

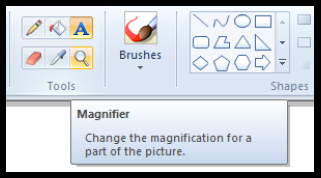
カテゴリ別メニュー



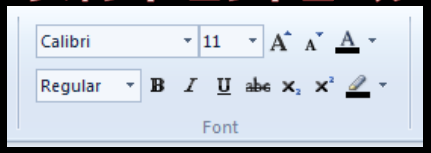
ドロップダウン ギャラリー



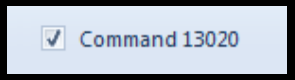
ツールチップ



フォント コントロール

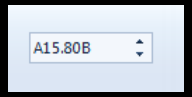


チェックボックス

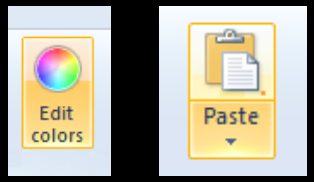


コンボボックス

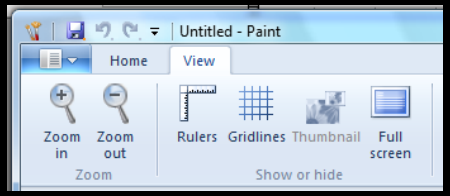
スピナー



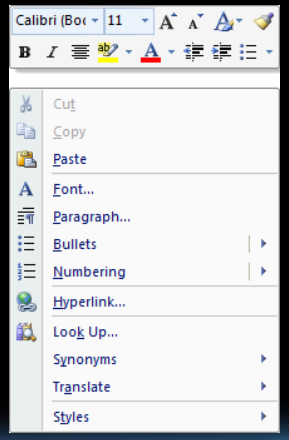
ボタン& スプリット ボタン



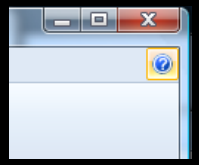
タブ &グループ



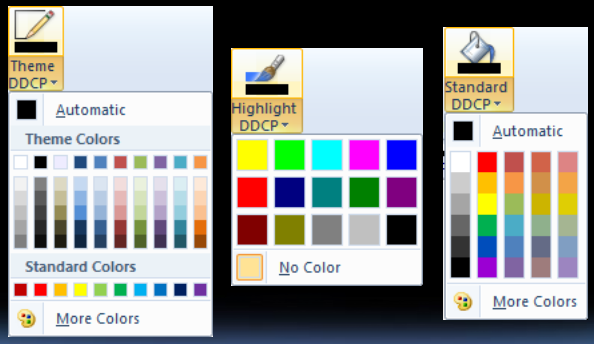
ミニ ツールバー& コンテキスト メニュー



ヘルプボタン

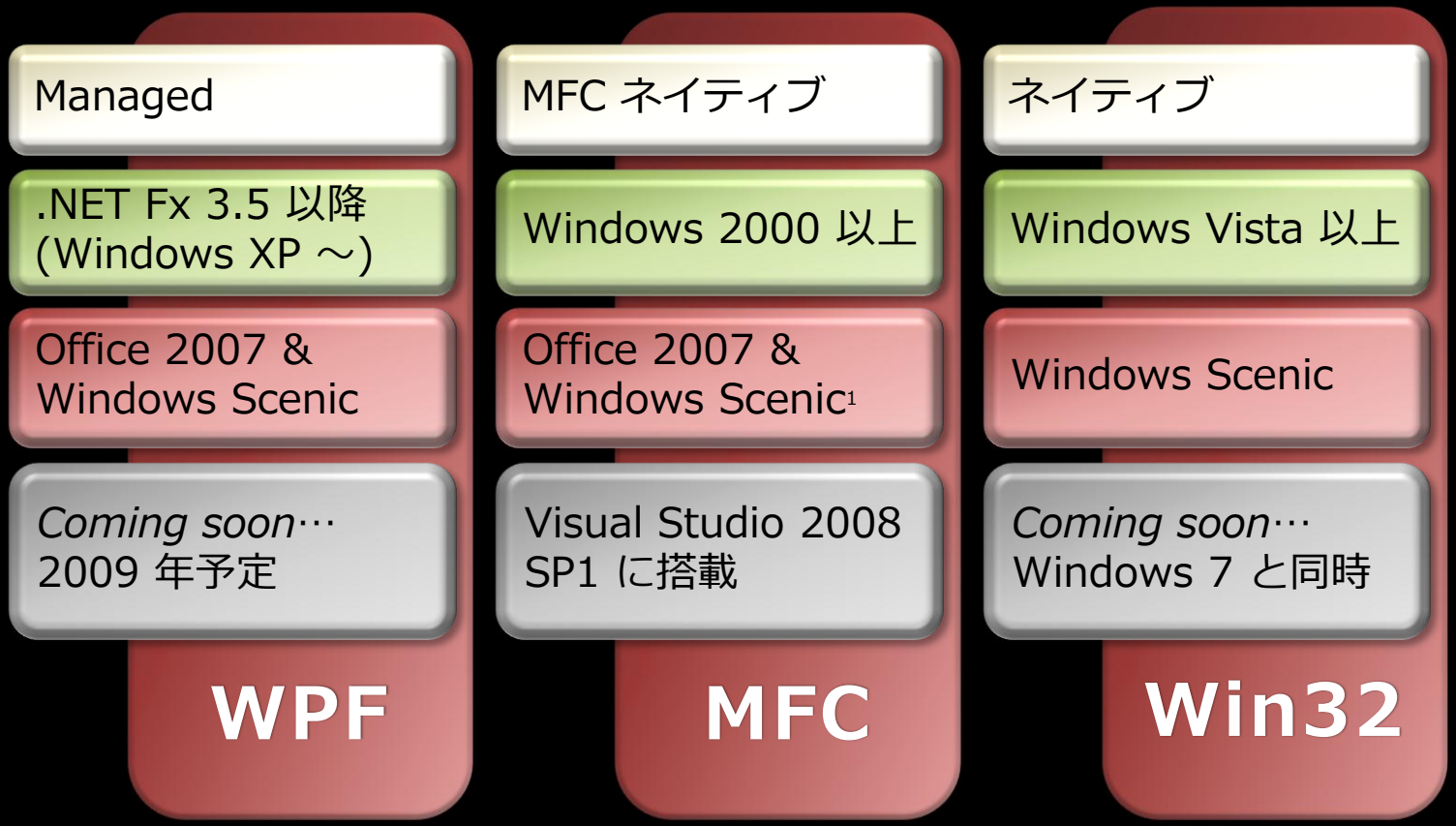


カラー ピッカー



リボン開発プラットフォーム

ソフトウェアベンダのさまざまなニーズに対応

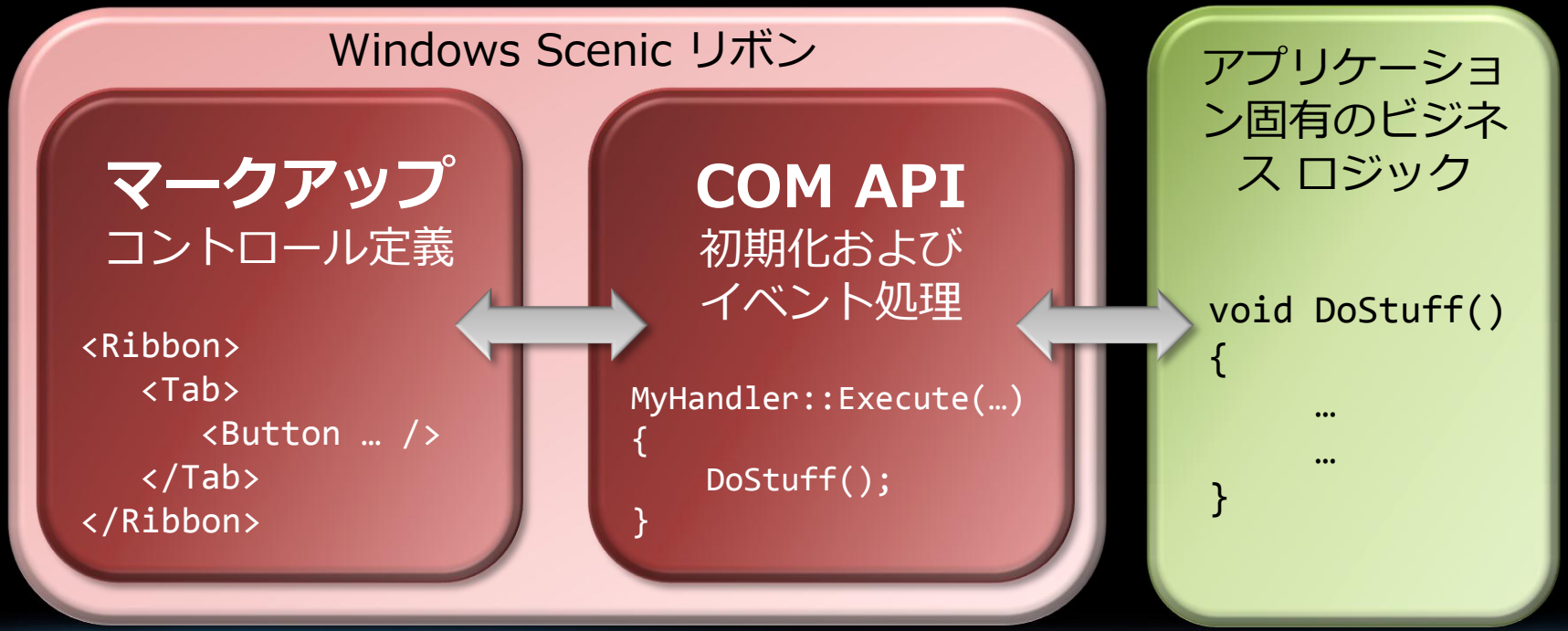


Microsoft の リボン Strategy

[1] *Office 2007* スタイル はリリース済み, *Windows Scenic* スタイル は *2009* 年を予定

Scenic リボン (Win32) 概要

- マークアップ ベースの UI
  - XML によるリボン コントロール定義
- 最上位のメニュー バー機能をオーバーライドし、リボン形式のコマンド UI をレンダリング
- COM API による初期化処理、およびコールバック メソッドの実装によるイベント処理

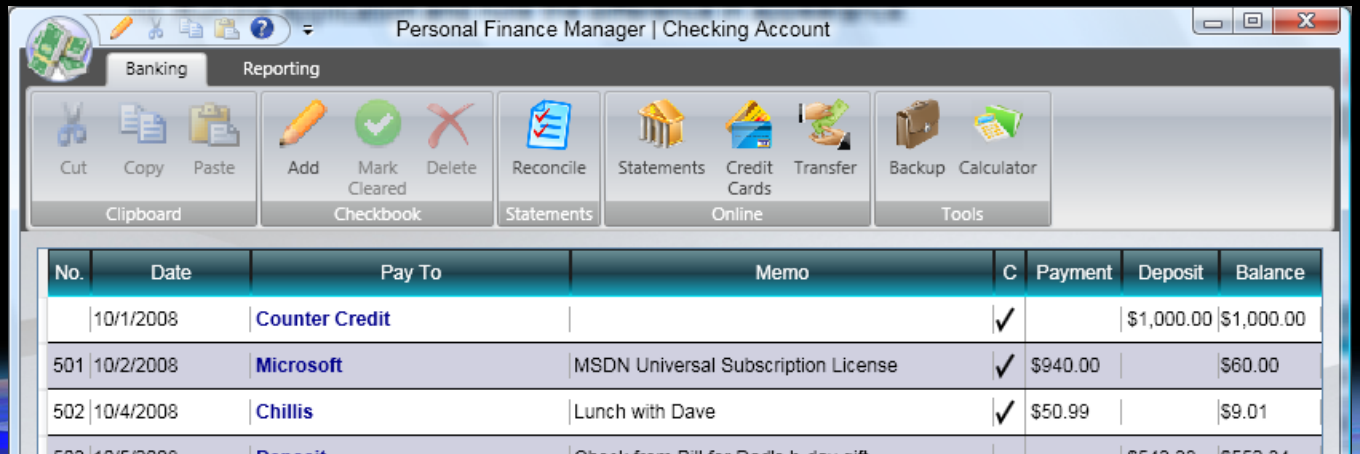


Scenic リボン

```
<Application xmlns="http://schemas.microsoft.com/windows/2009/Scenic/Intent">

  <Application.Commands>

    <!-- Clipboard commands -->
    <Command Name="Paste" Symbol="cmdPaste" Id="57637”>
      <Command.LabelTitle>
        <StringDef>Paste</Stringdef>
      </Command.LabelTitle>
      <Command.LargeImages>
        <Image Source="Paste.bmp"/>
      </Command.LargeImages>
    </Command>

  </Application.Commands>

  <Application.Views>
    <Ribbon Name="Microsoft.Scenic.Intent.RibbonSample">
      <Ribbon.Tabs>

        <Tab CommandName="TabHome">
          <Group CommandName="ChunkClipboard">

            <Button CommandName="Paste"/>
            <Button CommandName="..."/>
            ...

          </Group>
        </Tab>

      </Ribbon.Tabs>
    </Ribbon>
  </Application.Views>
</Application>
```

WPF リボン概要

- Office 2007 リボン仕様の .NET Framework 実装版
- .NET Framework 3.5 SP1 以上が必要
- XAML でリボン UI を記述
- 現在はプレビュー版として Code Plex で公開
  - WPF Ribbon Preview
    - http://www.codeplex.com/wpf/Wiki/View.aspx?title=WPF%20Ribbon%20Preview



WPF リボン

```
<r:RibbonWindow x:Class="WPFRibbon.MainWindow"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:r="clr-namespace:Microsoft.Windows.Controls.Ribbon;
           assembly=RibbonControlsLibrary">

  <DockPanel>

    <r:Ribbon DockPanel.Dock="Top"
              Title="{Binding RelativeSource={...},Path=Title}">

      <r:Ribbon.Resources>
        <r:RibbonGroupSizeDefinitionCollection x:Key="RibbonLayout"> ...
        </r:RibbonGroupSizeDefinitionCollection>
      </r:Ribbon.Resources>

      <r:RibbonTab Label="Tab 1">
        <r:RibbonTab.Groups>
          <r:RibbonGroup GroupSizeDefinitions="{StaticResource RibbonLayout}">
            <r:RibbonGroup.Command>
              <r:RibbonCommand LabelTitle="Cut" SmallImageSource="cut.png" />
            </r:RibbonGroup.Command>
            <r:RibbonButton Command="me:AppCommands.Cut"/>
          </r:RibbonGroup>
        </r:RibbonTab.Groups>
      </r:RibbonTab>

      <r:RibbonTab Label="Tab 2"> ... </r:RibbonTab>

    </r:Ribbon>

  </DockPanel>

</r:RibbonWindow>
```

Windows 7 : New APIs

demo

Scenic リボン (Win32)
Office 2007 リボン (MFC)
WPF リボン

まとめ

Windows 7 開発環境

- 現在、Windows 7 デスクトップは
  アンマネージド開発に対応
  - **Visual Studio 2008 SP1 + Windows 7 SDK**
- 今後の対応予定 …
  - **WPF** – .NET Framework 4.0 で登場予定
    - さまざまなプロパティを XAML で記述可能に
    - その他 WPF のメリット：binding, vector images
  - **MFC 7** – Visual Studio 2010 で登場
    - 新しいデスクトップ連携機能に対応した
      オブジェクトモデルをサポート
  - **Windows 7 Platform SDK**
    - Windows 7 Bridge – デスクトップ関連 API
      のためのマネージラッパー

Call to Action：
より自然なユーザー対話のために

- Windows 7 へのアプリケーション最適化
  - タスクバーとジャンプリストの使用
  - リボン UI の使用
  - マルチタッチ対応
  - 新しいグラフィック機能の活用
- 新しい API をチェック
  - Windows 7 Platform SDK
  - WPF 対応は .NET Framework 4.0 で

Windows® 7

Future Technology Days


Microsoft®