http://www.atmarkit.co.jp/fdotnet/dotnettips/234winxpstyle/winxpstyle.html
http://www.atmarkit.co.jp/fdotnet/dotnettips/235embmanifest/embmanifest.html

■マニフェスト・ファイルを使った Windows XP スタイルの実装

この方法は、.NET Framework 1.0（VS.NET 2002）のときから使用できるやり方であり、.NET Framework 1.1（VS.NET 2003）でも有効だ。

こちらの場合でも、事前に、次のコントロールの FlatStyle プロパティを「System」に設定しておく必要がある。

- Button コントロール
- GroupBox コントロール
- CheckBox コントロール
- RadioButton コントロール

この方法では、Windows アプリケーションに Windows XP スタイルの外観を持たせるのに、コードを書く必要はない。Windows アプリケーションと同じディレクトリに「マニフェスト・ファイル」を作成しておくだけで、そのアプリケーションに Windows XP スタイルの外観が適用される。

マニフェスト・ファイルは、アプリケーションが利用するコンポーネントなどの情報を記述するための XML ファイルである。アプリケーションに Windows XP スタイルの外観を持たせるには、この XML ファイルに Windows XP 用の新しい COMCTL32.DLL（Version.6 のコモン・コントロール）を使うように記述すればよい（古い Version.5 の COMCTL32.DLL がデフォルトで使われる理由を筆者は知らないが、恐らく Windows XP にネイティブ対応していないアプリケーションがそのまま動くように下位互換性を持たせるためだろう）。

具体的には、マニフェスト・ファイルとして次の内容のファイルを作成すればよい（テキスト・エディタなどで編集する）。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
<assemblyIdentity
  version="1.0.0.0"
  processorArchitecture="X86"
  name="Microsoft.Winweb.WindowsApplication1.exe"
  type="win32"
/>
<description>.NET control deployment tool</description>
<dependency>
  <dependentAssembly>
    <assemblyIdentity
      type="win32"
      name="Microsoft.Windows.Common-Controls"
      version="6.0.0.0"
      processorArchitecture="X86"
      publicKeyToken="6595b64144ccf1df"
      language="*"
    />
  </dependentAssembly>
</dependency>
</assembly>
```

Windows XP の外観を指定するためのマニフェスト・ファイルの内容

太字の部分（この例では「WindowsApplication1.exe」）にはアプリケーションの実行ファイルを指定するため、アプリケーションごとに記述内容が異なる。本稿の例では、アプリケーションが「WindowsApplication1.exe」というファイル名なので、このような記述となっている。

サンプル・ファイルのダウンロード（WindowsApplication1.exe.manifest）

このマニフェスト･ファイルの名前は､「＜アプリケーション名＞ .manifest」とする必要がある。例えば実行ファイルの名前が「WindowsApplication1.exe」というアプリケーションならば、マニフェスト・ファイルの名前は「WindowsApplication1.exe.manifest」となる。

■ isual Studio .NET を利用したマニフェスト・ファイルの組み込み

Win32 リソースであるマニフェスト･ファイルをアプリケーションに埋め込むには､Win32 リソース･ファイル（.res ファイル）を作成する必要がある。VS.NET でこれを作成するには、次の手順を実行する（作業手順は「VS.NET を使った Win32 リソース（マニフェスト･ファイル）の組み込み」の手順とほぼ同じ）。

１． VS.NET IDE のメニューから［ファイル］－［新規作成］－［ファイル］を実行して、［新しいファイル］ダイアログを開く。そのダイアログの［Visual C++］カテゴリの［テンプレート］の一覧から「リソース テンプレート ファイル（rct）」を選択してファイルを作成する。
２． リソース･エディタに表示されたリソース･テンプレート･ファイル（.rct ファイル。この例では「ResTempl1.rct」）を選択する。
３． リソース･テンプレート･ファイルが選択された状態で［プロパティ］ウィンドウの［Mfc Mode］プロパティを「False」に設定変更する。
４． リソース・テンプレート・ファイルを右クリックしてコンテキスト・メニューを表示し、そのメニューから［リソースの追加］を選択する。これにより、［リソースの追加］ダイアログが表示される。
５． ［リソースの追加］ダイアログの［インポート］ボタンをクリックすると、［インポート］ダイアログ（［ファイルを開く］ダイアログと同じ形式）が表示されるので、このダイアログで作成済みのマニフェスト･ファイル（本稿の例では「WindowsApplication1.exe.manifest」）を選択する。これにより［カスタム リソース の種類］ダイアログが表示される。［カスタム リソース の種類］ダイアログで、［リソースの種類］として「RT_MANIFEST」を入力して［OK］ボタンをクリックする。
６．「RT_MANIFEST」という種類のリソースとしてマニフェスト･ファイルがリソース･テンプレート･ファイルに追加される。その「RT_MANIFEST」リソースを選択して、［プロパティ］ウィンドウの［ID］プロパティを「1」に変更する。

以上で Win32 リソースの作成は完了だ。後は次の画面の手順で、これを Win32 リソース･ファイル（.res ファイル）として保存すればよい。ここで注意してほしいのは、必ず .res ファイルの形式で保存する必要があるということだ。次の画面でその手順を示す。

Win32 リソース・ファイル（.res ファイル）の保存
リソース･テンプレート･ファイル（.rct ファイル。この例では「ResTempl1.rct」）を Win32 リソース･ファイル（.res ファイル。この例では「WindowsApplication1.res」）として保存する。この［名前を付けてファイルを保存］ダイアログを開くには、VS.NET IDE のメニューから［ファイル］－［名前を付けて ＜拡張子を除くファイル名＞ を保存］（本稿の例では［名前を付けて ResTempl1 を保存］）を選択すればよい。

１． 保存する Win32 リソース・ファイルの名前（この例では「WindowsApplication1.res」）を指定する。
２． ［ファイルの種類］は「32 ビット リソース ファイル（*.res）」を選択する。デフォルトでは違う種類（「リソース テンプレート（*.rct）」）になっているので、間違えないように注意すること。