32bit用インストーラと64Bit用インストーラを同一のプロジェクトで作成する

注)このドキュメントは、InstallShield 2011 Premier Edition を基に作成しています。InstallShield 2011 以外のバージョンでは設定名などが異なる場合もあります。

概 要

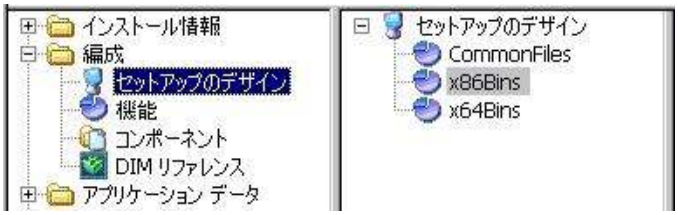
MSI 形式インストーラでは、Windows Installer の仕様により、32Bit環境と 64Bit環境の両方に対応したパッケージを作成することはできません。 32Bit用・64Bit用 にそれぞれ別々のパッケージとして MSI インストーラを作成する必要があります。

この記事では、単一のプロジェクトファイル(ism)を使用して、32Bit環境・64Bit 環境に対応した二つの MSI 形式ファイルを作成する場合の手順について説明します。

A. 32bit用機能・64bit用機能の作成

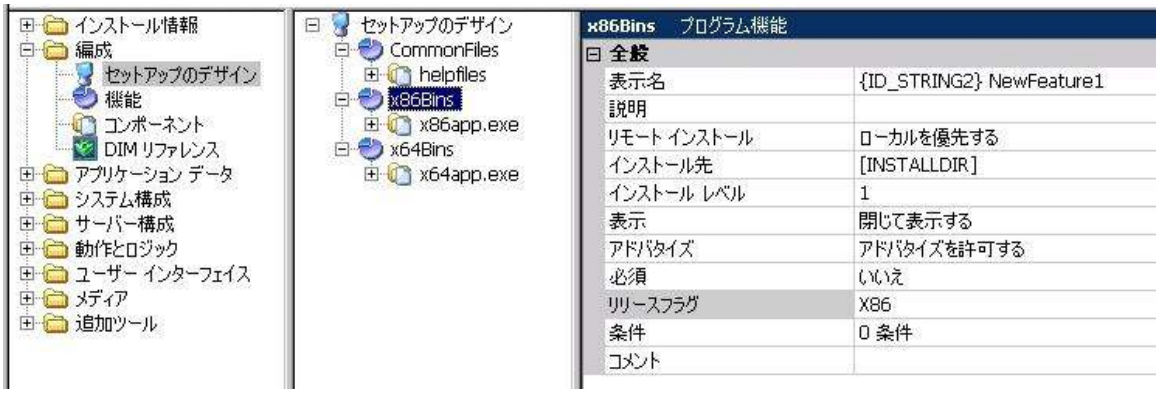
32bit用のファイルを含む機能と64bit用のファイルを含む機能をそれぞれ作成します。

1. [編成]ー[セットアップのデザイン]にて、二つの機能を作成します。片方の名前を[x86Bins] もう片方の名前を[x64Bins]に変更します。 両方の環境で同じ要素のインストールを行う場合、さらに三つ目の機能を作成します。(以下の画像では、[CommonFiles]という機能)



2. 作成した機能の配下に、任意のコンポーネントを作成してそれぞれの環境に準じたファイルを追加します。64bit バイナリを含むコンポーネントでは、コンポーネントの設定[64ビットコンポーネント]を[はい]に設定します

3. 機能［x86Bins］を選択します。右のビューにて、[全般]－[リリースフラグ]へ[X86]という文字列を設定します。



4. 機能［x64Bins］を選択します。右のビューにて、[全般]－[リリースフラグ]へ[X64]という文字列を設定します。

● リリースフラグについて

リリースフラグとは、インストーラのビルド時にインストーラへ含める要素をフィルタする機能となります。
リリースフラグが設定可能な項目としては、機能および InstallShield 前提条件 等があります。

32Bit・64Bit 環境にてそれぞれ、別々の InstallShield 前提条件をインストーラに含めたい場合は、[アプリケーション データ]－[再配布可能ファイル]ビューにて、該当の前提条件を右クリックして、[プロパティ]ダイアログにてリリースフラグの設定を行ってください。

設定例：

この画像の例では、X86・X64 環境用の前提条件がそれぞれ用意されているWindows Installer 4.5 の前提条件に対して、「X86」というリリースフラグを設定しています

※ なお、リリースフラグが空欄の機能・ InstallShield 前提条件は、すべてのリリースのビルド時に含まれます。

B. INSTALLDIR のリダイレクト

インストーラの標準インストール先となる INSTALLDIR の設定は、[インストール情報]－[一般情報]－[INSTALLDIR]にて行いますが、インストール先に C:¥Program Files を割り当てる場合、通常は以下の指定を行います。

X86 環境	[ProgramFilesFolder]
X64 環境	[ProgramFiles64Folder]

両方の環境に対応したプロジェクトを作成する場合、上記の指定を同時に行うことができないため、リリースフラグを条件に使用したカスタムアクションを作成する方法で対応を行います。

1. [インストール情報]－[一般情報]ビューにて [INSTALLDIR]へ [ProgramFilesFolder] を使用した 32Bit 環境用の階層を設定します。

2. [動作とロジック]－[カスタム アクションとシーケンス]ビューにて、[カスタムアクション]を右クリックして[新しいセットのプロパティ]を選択します。新規作成されたカスタムアクションを「Set64INSTALLDIR」とリネームして、右のウィンドウにて以下の設定を行います。

設定項目名	***設定内容***	***説明***
プロパティ名	INSTALLDIR	
プロパティ値	[ProgramFiles64Folder]Basic3264Sample	[ProgramFiles64Folder]を使用して任意の階層を指定
実行スケジュール	1 回のみ実行	
インストール UI シーケンス	<最初のアクション>	
インストール UI 条件	ISReleaseFlags><"X64"	ISReleaseFlags プロパティに X64 が含まれている場合
インストール 実行シーケンス	<最初のアクション>	
インストール 実行条件	ISReleaseFlags><"X64"	ISReleaseFlags プロパティに X64 が含まれている場合

3. 以下は設定後の画像となります

C. 製品構成・リリースの作成

テンプレート概要・リリースフラグの設定を行った製品構成を作成します。

1. [メディア]－[リリース]ビューにて、[リリース]を右クリックして、[新しい製品構成]を選択します。新規追加された製品構成を「x86 Configuration」とリネームします。同じように 64Bit 環境用の製品構成 「x64 Configuration」を作成します。

2. 製品構成「x64 Configuration」を選択します。右のウィンドウにて、以下の設定を行います。

   製品構成フラグ:「X64」

   テンプレート概要:「x64;1041」

3. 同じように製品構成「x86 Configuration」では、以下の設定を行います。

   製品構成フラグ:「X86」

   テンプレート概要:「Intel;1041」

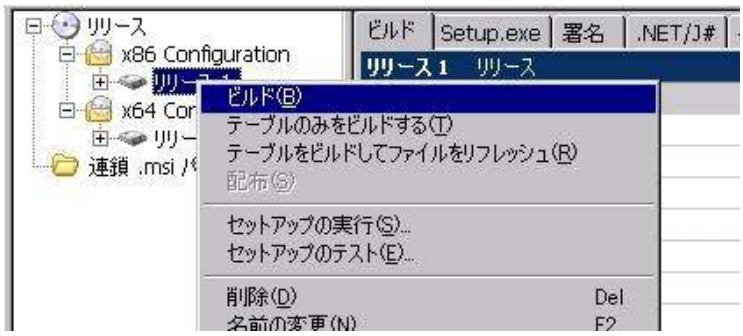
※ リリースフラグは大文字である必要性はありませんが、大文字と小文字は区別されます。機能に割り当てた文字列と製品構成で指定する文字列が同一になるようにしてください。

4. 製品構成「x64 Configuration」を右クリックして、[新しいリリース]を選択します。同じように製品構成「x86 Configuration」に対しても新しいリリースを作成します。

D. リリースのビルド・動作の確認

二つのリリースをビルドして、生成される MSI インストーラの動作を確認します。

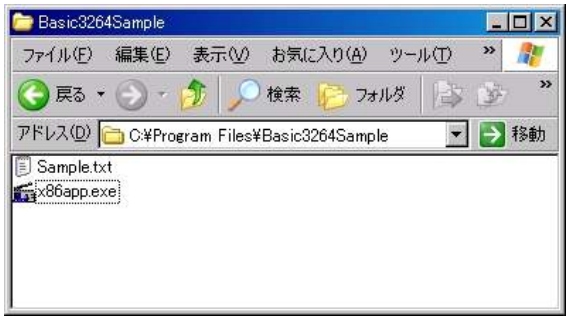
1. 製品構成の配下に作成されたリリースを右クリックして「ビルド」を選び、リリースのビルドを実行します。「x64 Configuration」「x86 Configuration」それぞれビルドを行ってください。



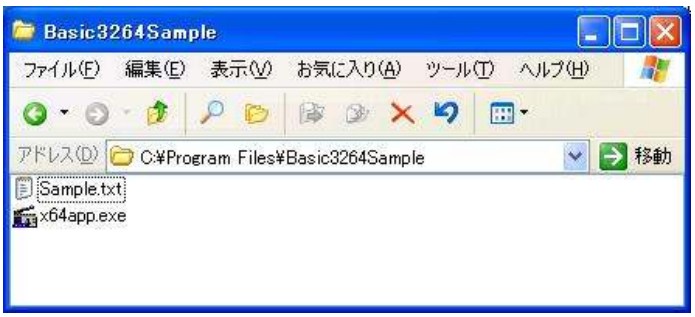
2. 生成された MSI インストーラの動作を確認します。「x64 Configuration」により生成された MSI ファイルは 64Bit 環境用 MSI として作成されるため、32Bit 環境では実行できません。

3. 32Bit 環境・64Bit 環境で生成されたインストーラをそれぞれ実行して、想定した箇所にファイルが転送されることを確認します。

32Bit 環境の例



64Bit 環境の例



補足: パス変数のオーバーライド設定を使用する（ カスタムアクション使用ファイルの置換 ）

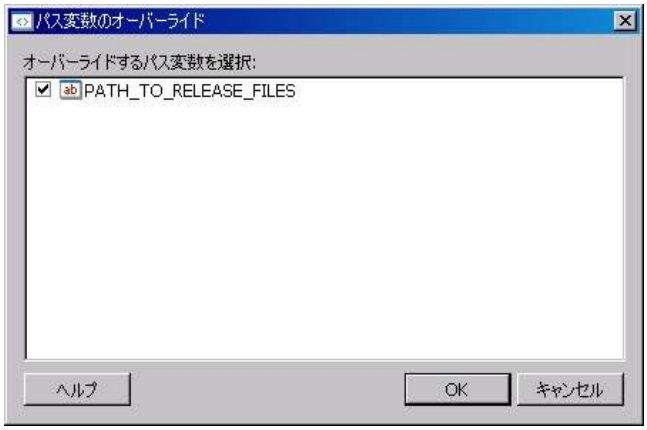
[リリース]ビューでは、[パス変数のオーバーライド]設定により、各リリースごとにビルドプロセスが参照するファイルの参照先階層を置換することが可能です。 例えば、階層 C:¥TEST_A を参照するパス変数 〈PATH_TEST_A〉 の参照先を、特定のリリースのみ C:¥TEST_B に置換することが可能です。
[パス変数のオーバーライド]設定は、例えば 32bit 環境用インストーラと 64Bit 環境用インストーラが異なるファイルを使用してカスタムアクションを実行しなければならない場合に有効な手段となります。

以下は、カスタムアクションが使用する DLL ファイルの参照先の置換を行う場合の手順例となります。

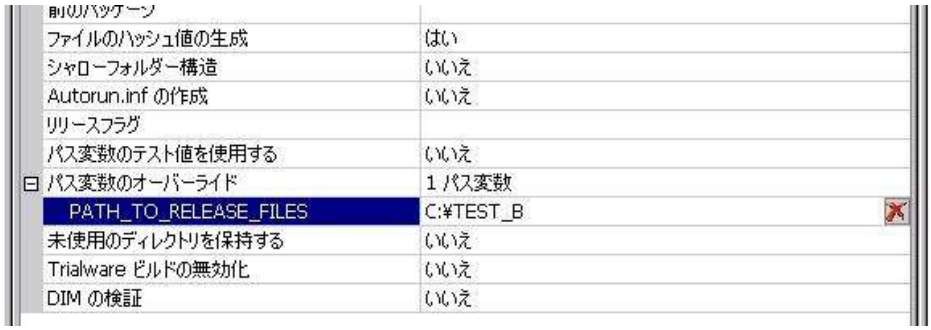
1. [動作とロジック]－[カスタム アクションとシーケンス]ビューにて、任意のカスタムアクションを選択します。右のウィンドウにて、カスタムアクションが DLL ファイルの参照に使用しているパス変数を確認します。
   （以下の画像では 〈PATH_TO_RELEASE_FILES〉）

2. [メディア]－[リリース]ビューにて、任意のリリースを選択します。右のビューの[ビルド]タブ－[パス変数のオーバーライド]の左端に表示される[…]ボタンを選択します。

3. 「パス変数のオーバーライド」ダイアログにて、任意の階層へ置換をするパス変数にチェックをつけます。



4. [パス変数のオーバーライド]設定配下に選択したパス変数に対するエントリが生成されますので、置換を行う別の階層を指定します。



5. 設定を行ったリリースをビルドした場合、該当のパス変数は置換された階層に保存されているファイルを参照します。生成される MSI インストーラはデフォルトのビルド時と異なったファイルを保持します。

※ 注意: この設定を使用することで、リリースごとに異なるファイルを含めることが可能になります。しかしながら、MSI 形式インストーラでは、通常 64Bit バイナリを含むコンポーネントには [64 ビットコンポーネント]設定を行う必要があるため、32Bit・64Bit 環境で異なるファイルを転送する必要がある場合でも、この設定を使用する方法は推奨されません。