

Creating a Complete Pocket PC Installer: Using NSIS & Astrum Install Wizard

Author: Rahul Gaur
Mindfire Solutions, www.mindfiresolutions.com

Table of Contents

Overview	3
Target Audience	3
Conventions	3
1. Introduction	4
2. Tools & SDKs Used	4
3. Creating Pocket PC Installer	5
3.1. MainAppPPC	5
3.2. MainAppPPCSetup	10
4. Conclusion	14

Overview

“*What is a User’s First Experience with a Product?*”

Yes, you guessed it right. It’s an **Installer**...

A stable and reliable installer is an important component of a successful software product. Things become more typical when you are talking about software for handheld devices like pocket pc, palm, etc.

Via this document, we are putting forth our experience with PocketPC installer, with an aim to help our developer community get a quick and trouble free Windows Installer for a Pocket PC Application(s).

The document does not discuss creating Pocket PC Application setup packages (for which we can use Cab Wizard utility freely available) but how to include multiple Pocket PC CabWiz files, having different applications, into one package and then integrating the same with any Third-Party Proprietary Windows Installer to give the user a seamless installation experience.

Target Audience

This document is intended for Microsoft Pocket PC & Smartphone application developers. The reader should already be familiar with WinCE Tools like CabWizard-Cabwiz.exe, Microsoft ActiveSync etc., and are aware of INF¹, INI¹, CAB¹ Files & Setup.dlls² with respect to installation of software on pocket pc side. We will not be going into such details.

Conventions

Throughout this document, any **important text** is in Bold Times New Roman Font. The **code snippets** are in Courier New Font with *code comments* in italics

¹ <http://msdn2.microsoft.com/en-us/library/aa448616.aspx>

² <http://www.pocketpcdn.com/articles/setupdll.html>

1. Introduction

Creating an Installer for a software product is an art in itself. Essentially one has to take care of the following three steps:-

- Fresh Installation
- Uninstallation
- Reinstallation

At every step one has to be aware of the changes required on the target machine(s) and never loose the focus to keep installation experience as painless as possible for the user. Creating a Pocket PC or a Smartphone Windows Installer is no exception

Various components which can be part of a full pocket pc installer are:-

- Application(s) that need to be installed on Pocket PC device.
- Application(s) to be installed on PC, and
- ActiveSync Service Providers(Conduits)

It does become a little bit tedious with all the things involved in same installer, but there is no way out. You need to take the bulls by its horns.

In this document we discuss, in detail, a Windows Installer made for a sample Pocket PC project. The project comprises of the following binaries:-

- a) MainApp.exe: The Main Pocket PC Application
- b) AppAlarm.exe: An Alarm Application for Pocket PC. It supplements the MainApp.exe
- c) DevSyncApp.dll: Device Side ActiveSync Conduit of MainApp.exe
- d) SyncApp.dll: Desktop Side Sync Conduit of MainApp.exe
- e) MainAppSetupDll.dll: It is a setup dll for MainApp that enables us to do some custom actions during the installation or un-installation on device side.

2. Tools & SDKs Used

- a) Microsoft Embedded Visual C++ with Service Pack 3
- b) Microsoft Visual C++ 6.0
- c) Microsoft Pocket PC 2003 SDK
- d) Microsoft ActiveSync 4.0.0
- e) Astrum InstallWizard v2.24 (Evaluation Version)
(<http://www.thraexsoftware.com/aiw>)
- f) Nullsoft Scriptable Install System(NSIS) v2.21
(http://nsis.sourceforge.net/Main_Page)

3. Creating the Pocket PC Installer

Our project has two components:-

a) MainAppPPC.exe

A setup executable created using NSIS. It contains the files MainApp.ARMV4.cab, MainApp.ini, AppAlarm.ARMV4.cab and AppAlarm.ini. MainApp.ARMV4.cab & AppAlarm.ARMV4.cab are the Cabinet Files created using CabWizard and the MainApp.ini & AppAlarm.ini are their respective INI Files. The first cabinet file comprises of MainApp.exe, SyncApp.dll & MainAppSetupDll.dll and the second one comprises of AppAlarm.exe only

b) MainAppPPCSetup.exe

It is a Windows Installer created using Astrum InstallWizard. One can think of it as a Setup Container having MainAppPPC.exe, SyncApp.dll and Setup Instructions etc.

Details follow....

3.1 MainAppPPC.exe

Before discussing this executable in detail we will first discuss about MainAppSetupDll.dll. We need to introduce two functions in the dll, **StartActiveSync()** & **EndActiveSync()**.

As the names suggest, StartActiveSync() starts the ActiveSync & EndActiveSync() ends the ActiveSync on the Device Side. These functions are important because when the device is in Standard Partnership with ActiveSync with our conduit “**enabled**”, these functions will help us break and make the partnerships again with the Device and also will enable us to do any cleanup required Without these functions we will face problems with un-installation (DevSyncApp.dll may not get deleted) and reinstallation (DevSyncApp.dll may not get replaced) processes.

EndActiveSync() should be called at the starting of **Install_Init** & **Uninstall_Init** whereas StartActiveSync() should be called at the starting of **Install_Exit** & **Uninstall_Exit**. The code snippets are given below:

```
void StartActiveSync()
{
    HWND hWnd = NULL;
    hWnd = ::FindWindow(NULL, L"ReplLog");

    if(!hWnd)
    {
        PROCESS_INFORMATION pi;
        CreateProcess( L" \\windows\\repllog.exe", NULL, NULL
                    , NULL, NULL, CREATE_NEW_CONSOLE
                    , NULL, NULL, NULL, &pi
                    );
    }
}
```

```
void EndActiveSync()
{
    HWND hWnd = NULL;

    hWnd = ::FindWindow(NULL, L"ReplLog");
    if(hWnd)
    {
        ::PostMessage(hWnd, WM_CLOSE, 0, 0);
    }
}
```

Lets move ahead and get into the details of MainAppPPC

As discussed earlier it is created using NSIS. NSIS is an open source script-based system to create Windows Installer. We assume that NSIS is installed in its default directory - **C:\Program Files\NSIS**.

Create a folder by the name of **MainAppPPC** in the **Examples** Folder of NSIS.

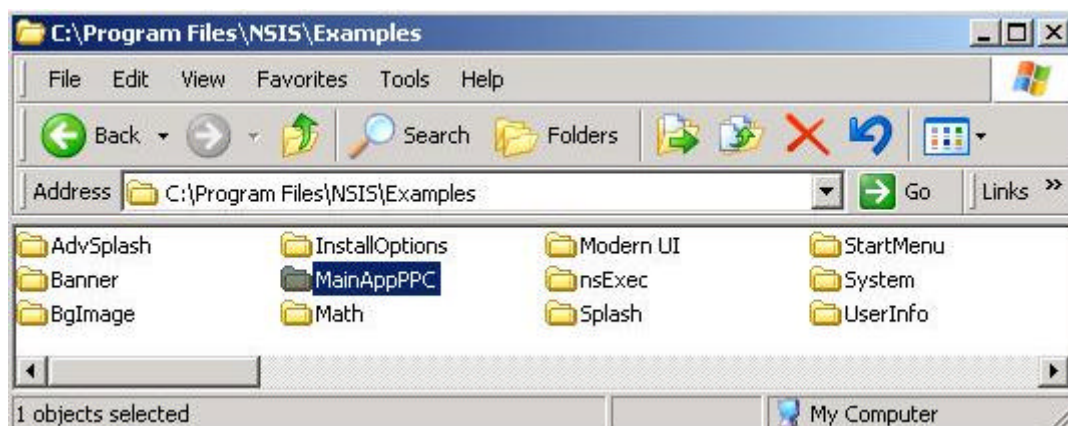


Figure 1: Creating a MainAppPPC Folder

Copy the following files into this folder

- MainApp.ARMV4.cab
- MainApp.ini
- AppAlarm.ARMV4.cab
- AppAlarm.ini
- MainApp.ico (Icon File)

Now lets look at the NSIS installer script. Create a text file by the name of MainAppPPC.nsi (*.nsi files are the NSIS Script Files). Readme.txt & Eula.txt are not the part of this package. These will be included in the Astrum Install Wizard Project and will be covered in the next section.

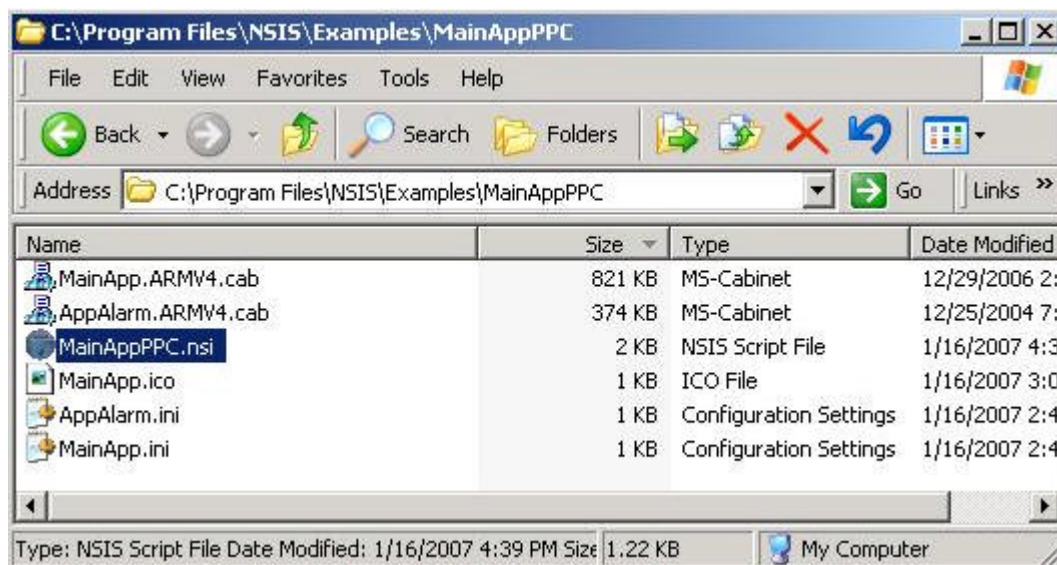


Figure 2: Contents of MainAppPPC Folder

The contents of the MainAppPPC.nsi is listed below

```
Name      "MainApp"
Icon      "MainApp.ico"
OutFile   "MainAppPPC.exe"

;Directory to which it extracts the cab
InstallDir "$TEMP \MainAppPPC"

Section   "MainApp" ; (default, required section)
SetOutPath "$INS TDIR"

File MainApp.ini
File MainApp.ARMV4.cab
File AppAlarm.ini
File AppAlarm.ARMV4.cab
```

```
;One-time initialization needed for InstallCAB subroutine

ReadRegStr $1 HKEY_LOCAL_MACHINE
"software \Microsoft \Windows \CurrentVersion \App Paths \CEAppMgr.exe "
" "

IfErrors Error
Goto End
Error:
MessageBox MB_OK|MB_ICONEXCLAMATION \
"Unable to find Application Manager for PocketPC applications. \
Please install ActiveSync and reinstall YourApp." \
End:

;Install pocket pc cabs

StrCpy $0 "$INSTDIR \AppAlarm.ini"
Call InstallCAB

Sleep 2000

StrCpy $0 "$INSTDIR \MainApp.ini"
Call InstallCAB

SectionEnd ; end of default section

;(post install section, happens last after any optional sections)
Section "-post"
Quit
SectionEnd ; end of -post section

; Installs a PocketPC cab-application
; It expects $0 to contain the absolute location of the ini file
; to be installed.
Function InstallCAB
ExecWait "$1" "$0"
FunctionEnd

; EOF
```

Most of the Script Functions are self-explanatory. Please refer the NSIS help file i.e. NSIS.chm for more details).

Briefly one can say that the script starts with the name, icon, outfile followed by all the files which constitute this package and which are actually extracted into

InstallDir.

To install a CAB File on the Pocket PC, this setup application has to start CeAppMgr.exe, passing as parameter the complete path of the INI File. To accomplish this, this setup executable reads the location of CeAppMgr.exe from the registry

```
ReadRegStr $1 HKEY_LOCAL_MACHINE
"software \Microsoft \Windows \CurrentVersion \App Paths \CEAppMgr.exe"
"
```

Installing a CAB now becomes as simple as

```
StrCpy $0 "$INSTDIR \AppAlarm.ini"
Call InstallCAB
```

where InstallCAB is :-

```
Function InstallCAB
ExecWait "$1" "$0"
FunctionEnd
```

We have purposefully introduced a delay of 2 seconds between two consecutive calls to InstallCAB function so that user does see two installation instructions coming up at separate times.

Two points are noteworthy:-

- a) The First call to InstallCab should be with AppAlarm.ini because, as discussed earlier, we are ending the ActiveSync session on the device side during the Install_Init method(in MainAppSetupDll.dll) of MainApp.ARMV4.cab. In case the first call is with MainApp.ini then the installation of AppAlarm.ARMV4.cab will go into the Pending Installation due to ActiveSync session not there which is required for any installation on Pocket PC
- b) There are many Professional Pocket PC/Smartphone installers which can combine multiple cabs into one Windows executable, but the reason why NSIS is so popular is that it is an open source freeware and also if the ActiveSync gets terminated on the device or desktop side then the subsequent Cab-Installations goes into a pending state. When the device is connected again these installation restart. This feature is unavailable in many of the other Installers we tried

Now that the script MainAppPPC.nsi is over, right-click on it and choose the option **Compile NSIS Script**. The NSIS compiler will run and compile the script to produce MainAppPPC.exe. Now we can integrate it with SyncApp.dll (desktop side ActiveSync dll) using Astrum Install Wizard to get a complete installer.

3.2 MainAppPPCSetup.exe

Note: We assume that the Astrum Install Wizard is installed in its default directory i.e. C:\Program Files\Astrum InstallWizard 2

- 1) First create a folder by the name of **MainAppPPCSetup** in C:\ Drive.
- 2) Copy MainAppPPC.exe & SyncApp.dll also into this folder.
- 3) Double click on “**aiw2.exe**”. You may be presented with a help screen depicting the various universal constants & declarations used in Astrum Install Wizard projects. One quick glance would be enough to get started.

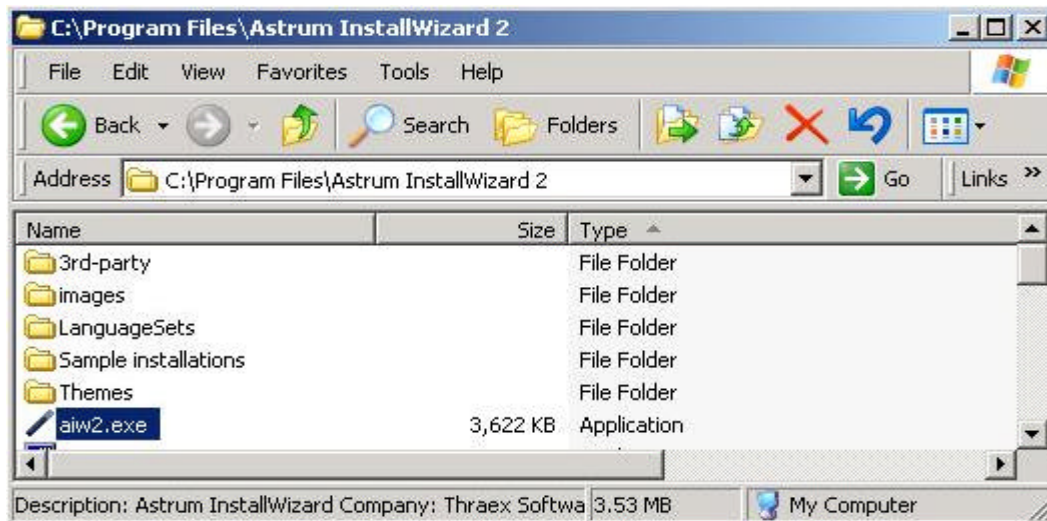


Figure 3: Selecting aiw2.exe

- 4) Once we are through with it, select “**Blank Setup**” from the Project Wizard screen.
- 5) Go to File Menu and select “**Save Project As**” option. Name the file as “**MainAppPPCScript**” and save it in “C:\MainAppPPCSetup” folder.

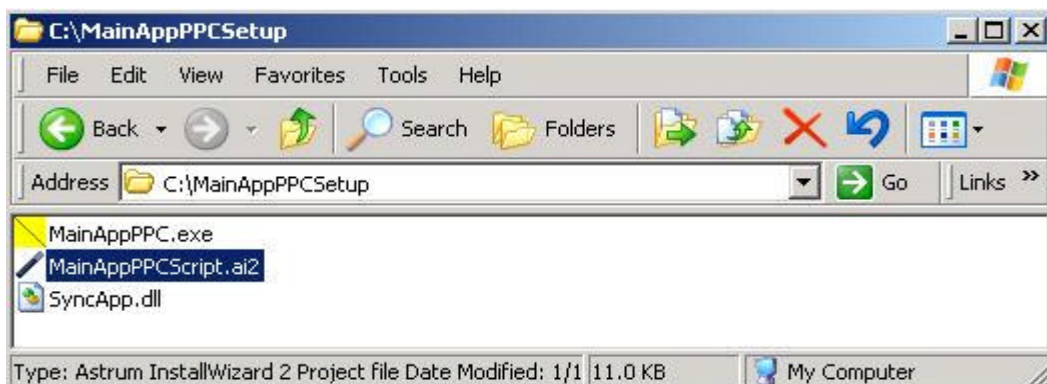


Figure 4: Contents of MainAppPPCSetup Folder

MainAppPPCScript.ai2 is the script file for Astrum installer. Yes this is another script file, but don't worry this is one of the easiest and most user friendly installer script we have seen. When you double click on it, it opens the Astrum Install Wizard for our project. There are various links on the left hand side. These are actually all the steps, listed in sequence that are needed to build the installer. Number of steps may overwhelm a newbie but believe me it hardly takes few minutes to get going. The steps are discussed below:

- a) **General:** Here one can fill the Name of Application (MainApp), Name of Company, Version No.(1.00), Default Installation Path etc.
- b) **Dialogs:** These are the various dialogs that run in sequence during installation. The default script has the Welcome, Select Destination Directory, Select Shortcut Folder, Summary, Installing, Finished & Uninstall Dialogs selected. One can check out the License Agreement & Readme dialogs also as per the need of the project. Double clicking on any dialog opens up Astrum Dialog Editor, using which one can change the dialog template also. Very convenient to use.
- c) **Language:** For different languages support
- d) **License:** For adding license related stuff
- e) **Readme:** For adding readme text to be shown to user after installation is over.
- f) **System Info:** For retrieving and showing the System Information like available memory etc.
- g) **User Info:** For querying the user's information and for generating the serials.
- h) **Options:** It has some features related to Options, Install & Finished Dialog
- i) **Uninstall:** This is for adding an entry into Add-Remove Programs, for running a custom executable (like cleanup.exe etc.) before the start of uninstallation and for creating an Uninstaller.exe
- j) **Files to Install:** It is an important section which contains all the files & the setup types, **Minimal, Custom or Typical**. Select Application Files in the installation tree and click on "Add Files" button and select MainAppPPC.exe & SyncApp.dll. Remember that SyncApp.dll is a desktop side ActiveSync Service Provider. It needs to be registered on the user's PC. So select SyncApp.dll and click on **Advanced** button. Check the "Shared component" & "Register ActiveX control/ Type Library" checkboxes and click on the OK Button.

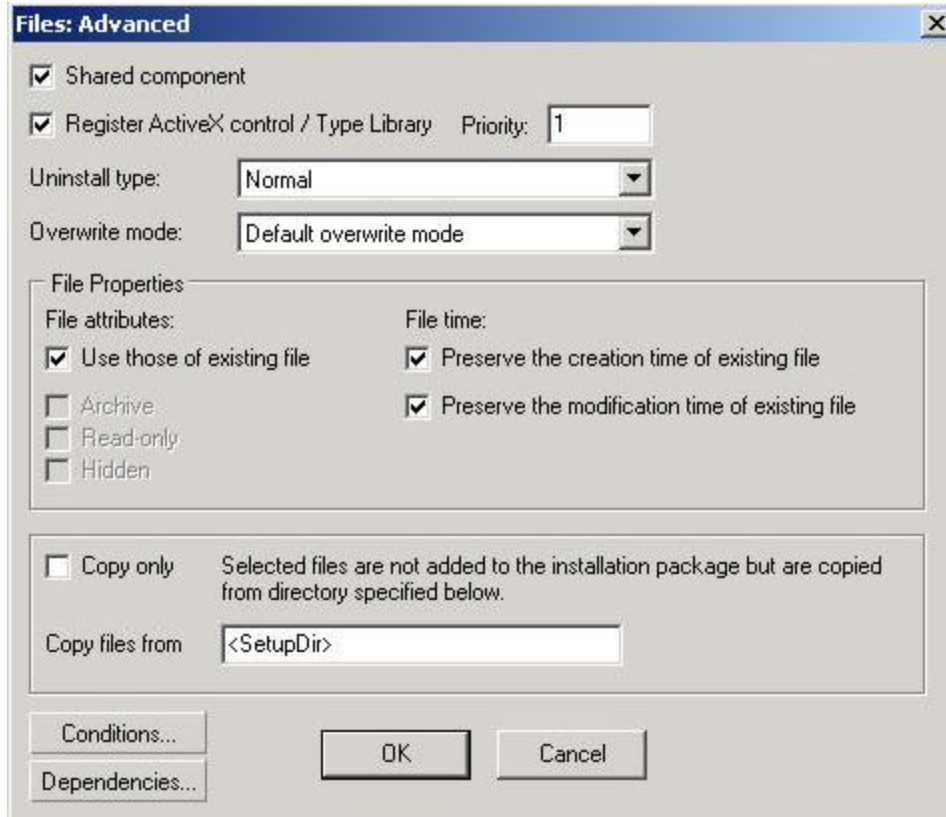


Figure 5: Settings for Registering SyncApp.dll

- k) **System Changes:** For specifying which, if any, third-party components to install to the destination system. In our case there is none.
- l) **Registry:** The script already has some registry entries in the subkey HKEY_LOCAL_MACHINE\Software\<Company Name>\<AppName>
No more registry entries are required in our case.
- m) **Shortcuts:** The script already has a shortcut for Uninstaller.
- n) **Ini Files, Text Files & File Associations:** These steps are related to the System Changes and there are no system changes required for this project.
- o) **System Requirements:** For checking the destination system. One can change the settings here as per the desired properties.
- p) **Background:** For customization of the outlook of the installation background window. No changes required here.
- q) **Slide Show:** It is a feature for big installation. No changes required here also since ours is small package.

- r) **Advanced:** It has some advanced settings. Make sure that “**Check for previous installations**” checkbox is checked so that on **re-installation**, the previous version can be uninstalled. The previous installation is generally checked from the registry entries in **Registry** step.
- s) **Variables:** For specifying our own variables that can be used exactly as the reserved keywords. For this project we don't require any variables.
- t) **Interactive and Shell:** It is for adding interactive operations like execute some programs, open a document etc. and for adding standard file operations. It is an important step because we have to run MainAppPPC.exe and hold the installation and once the installation completes, delete MainAppPPC.exe. So its one **Interactive Operation** and one **File Operation** for us. Under “**Interactive operations**” section click on the “**Add**” Button and fill the various fields as shown below.

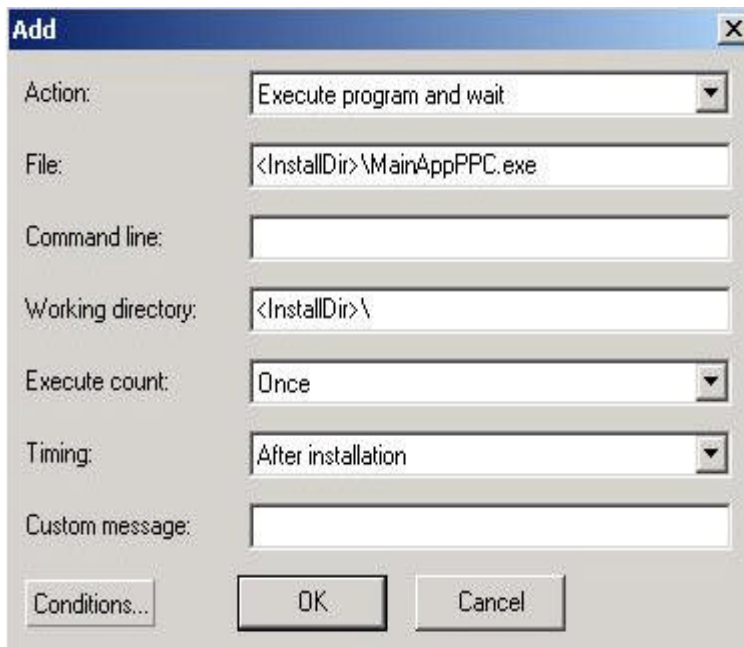


Figure 6: Interactive Operation for MainAppPPC.exe

Then click on the “**OK**” Button.

Under “File operations” section click on the “Add” Button and fill the various fields as shown below:

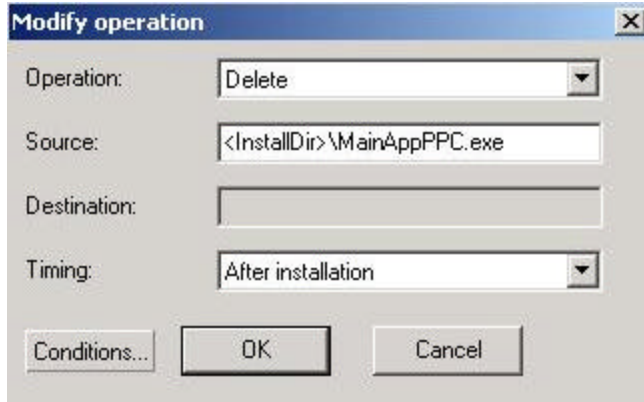


Figure 7: File Operation for MainAppPPC.exe

Then click on the “**OK**” Button

- u) **Resource Files:** No change
- v) **AutoUpdate:** No change.
- w) **Create Setup:** Yes, we have reached the final step. In the “**Media options**” section put the complete output file name in the “**Setup filename:**” edit box which is “**C:\MainAppPPCSetup\MainAppPPCSetup.exe**”.

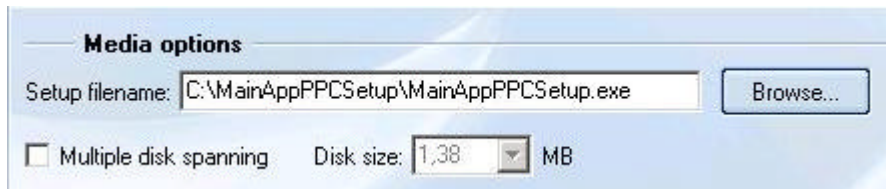


Figure 7: Naming Our Setup File Name

Hit the “**Create Setup**” Button and the “**MainAppPPCSetup.exe**” gets created in “**C:\MainAppPPCSetup**” Folder. One can view the steps in “**Status output**” window.

Yes!!, its done. One can test this installer and add other features as and when required.

4. Conclusion

Hope you found a quick and a convenient way of making a complete Pocket PC Installer having multiple cabs, ActiveSync Service Providers and also PC components. If you have any questions, please contact us and we would be glad to help you out.

===== THE END =====

Author of this article is associated with Mindfire Solutions (www.mindfiresolutins.com), which is an offshore software development company and provides customized software solutions to global clients.