WireLess Deployer

Package Reference Manual

Version 1.3

Revision History

Changes to the original manual are listed below:

Version	Date	Description
1.0	08/29/2008	Initial release.
1.1	09/16/2008	Action-Regexp and Action-Attrib
1.2	09/15/2009	Environment Variables.
1.3	08/26/2010	Delete reg entries

Table of Contents

Overview

This documentation describes the files involved on package download with WireLess Deployer:

- **Package descriptor** file (Id-Version.pckg)
- **Installation** file (the name of this file varies).
- **Uninstallation** file (the name of this file varies).

These files can be modified using Packet Express, or they can be modified by hand (not recommended). Current version of Packet Express allows building **only application** packages.

Package Descriptor

This file –located on "`X:\Program Files\WireLessDeployer\Packages`"– contains information about the package.

The name of this file is composed of the Id and version number, as follows:
Id-Version.pckg

"*Id*" and "*Version*" **must** match the Id and Version of PACKAGE section, otherwise the package will be invalid.

There are two kinds of packages: application and configuration packages.

- An **Application** package descriptor contains PACKAGE, INFO and APPLICATION sections.
- A **Configuration** package descriptor contains PACKAGE, INFO and CONFIG sections.

PACKAGE section:

Type

Type = A | C

A = application
C = configuration

Example:
Type = A

Id

Identifier.

The combination Id-Version identifies a package univocally. This ID is also the file and folder name of the package. It is MANDATORY that this ID contains only "legal" Windows file and folder characters.

Version

Version = Major-minor-revision.

Only digits, separated by dashes "-".

Example:
Version = 1-2-3

Name

Name of the package. In application packages, the name of the package is displayed to user on the WDp Console and on the WDp Launcher.

FileList

The installation file is specified here. This file contains a list of files to be downloaded and actions to be executed (see **FileList**). The FileList is requested when the package is being installed.

Example:
FileList = install.lstdnl

UninstallList

The uninstallation file is specified here. This file may contain a list of files to be downloaded and actions to be executed (see **FileList**). The UninstallList is requested when the package is being uninstalled.

Example:
FileList = uninstall.lstdnl

Persistent

persistent = yes | no

This option specifies whether the package information persists after a coldboot. In order to make the package files persistent, the package content **must** be stored in a cold-boot **persistent** Flash folder. Use ACTIONS, like [ACTION-CPY] to copy them into a flash folder (see **Actions Reference**), or install the files into the persistent flash folder directly.

INFO section:

This section is purely informative and is used for displaying package information in the WDp Console. This means that neither of these fields will be used on the PDA.

Description

Description of the package. This key is for information purposes only.

Platform

WINCE410 | WINCE420 | WINCE50 | WINCE2003 | WINCE2005

Specifies the Platform Id in which the package will be installed.

WINCE410: Windows CE 4.10
WINCE420: Windows CE 4.20
WINCE50: Windows CE 5.0
WINCE2003: Windows Mobile 2003 / Pocket PC 2003
WINCE2005: Windows Mobile 5.0

WDp Console and Agent will use the Platform value in Collections and Formulas to evaluate package download conditions.
It is recommended, but not mandatory, that this information matches the real operating system version of the PDAs that will download the package. The Client doesn't check this value.

Example:
Platform = WINCE50

Model

Specifies the model name or standard name of the PDA.

WDp Console and Agent will use the Model value in Collections and Formulas to evaluate package download conditions.
It is recommended, but not mandatory, that this information matches the real terminal model of the PDAs that will download the package. The Client doesn't check this value.

Example:
Model = mc9090

Vendor

Specifies who made the package. This key is for information purposes only.

Example:
vendor = Softogo SA

APPLICATION section:

This section is optional and is used only in application packages executed by the Launcher.

StartExe

Specifies the complete path to a shortcut or an executable file on the PDA.
See [ACTION-EXE] section of **WDP files** chapter for an explanation of this option.

The icon displayed by the Launcher is extracted from this executable file. If no icon is available or the file is not an exe, a standard icon will be displayed.

Example:
StartExe = \windows\calc.exe

CONFIG section:

This section is optional and is used only in configuration packages. It allows user to modify some package files in the WDp Console. The files in this list are not downloaded on the PDA.

Filelist

Specifies a configuration list file.

Example:
Filelist = config.lstcfg

An Application Package Descriptor example:

```
# APP_SETUP-1-0-0.pckg

[PACKAGE]
type=A
id= APP_SETUP
name= APP setup
version=1-0-0
filelist=install.lstdnl
uninstalllist=uninstall.lstdnl
persistent=yes

[INFO]
description= APP application setup
platform=WINCE2005
model=MC9090
vendor=Softogo Packet Express

[APPLICATION]
StartExe= \windows\app.exe
```

A Configuration Package Descriptor example:

```
# APP_CONFIG-1-0-0.pckg

[PACKAGE]
type=c
id= APP_CONFIG
name= APP configuration
version=1-0-0
filelist=install.lstdnl
uninstalllist=uninstall.lstdnl
persistent=yes

[INFO]
description= APP configuration
platform=WINCE2005
model=MC9090
vendor=Softogo Packet Express

[CONFIG]
filelist = config.lstcfg
```

FileList

This file defines a list of files to be downloaded and a set of actions to be executed after downloading process.

The file is divided into three sections: MAIN, DOWNLOAD and ACTIONS.

MAIN section:

[main]
sequentialActions = yes | no

SequentialActions option allows executing the actions as they were written. If this option is not active or not specified the actions will be executed following a predetermined order (see ACTION sections).

DOWNLOAD section:

Defines which files will be downloaded.

[download]
filecount = number of files (NN+1)
file00 = source file > target file
…
fileNN = source > target

Source and Target must be separated by a ">".

Source: Path of the file in the Server.
Target: Complete path of the file in the PDA.

Quotation marks are not required for source and target.

ACTION sections:

These sections define a set of actions to be executed after downloading process. Supported actions are:

ACTION-ZIP, ACTION-CPY, ACTION-ATTRIB, ACTION-REG, ACTION-DLL, ACTION-EXE, ACTION-REGEXP, ACTION-DEL, ACTION-REMOVE, ACTION-KILL, ACTION-DELAY and ACTION-BOOT.

See **Actions Reference** for a detailed description of each action.

There are two modes for processing actions: non-sequential and sequential.

In non-sequential mode, actions are processed in the following order:

1) ACTION-ZIP
2) ACTION-CPY
3) ACTION-ATTRIB
4) ACTION-REG
5) ACTION-DLL
6) ACTION-EXE
7) ACTION-REGEXP
8) ACTION-DEL
9) ACTION-REMOVE
10) ACTION-KILL
11) ACTION-DELAY
12) ACTION-BOOT

To use this mode, deactivate *sequentialActions* option in MAIN section.

In sequential mode actions are processed as they were written, except for ACTION-BOOT, which is executed always at the end, no matter its position on the file. In order to use this mode you have to activate *sequentialActions* option in MAIN section.

Unit Variables

There is a set of variables that may be used in the DOWNLOAD section of a FileList. These variables characterize a Unit.
The variables are:

%HOSTNAME%
%IP%
%MAC%
%MUID%
%UID%
%MODEL%
%PLATFORM%

The client searches the DOWNLOAD section for these variables replacing them by its real value. The search is performed on the "source" part of that section, that is, the remote file path.

Assuming a Unit has these variables,

HOSTNAME = PDA001
IP = 192.168.1.9
MAC = AA:BB:CC:DD:EE:FF
MUID = ABC123DEF2345
UID = john
MODEL = MC9090
PLATFORM = WINCE50

and a Package FileList contains these files,

```
[DOWNLOAD]
filecount  = 4
file00     = PDA\App\%MODEL%\app.exe > \Application\App\app.exe
file01     = PDA\Cfg\%HOSTNAME%.cfg  > \Application\App\app.cfg
file02     = PDA\%UID%\file1.txt     > \Application\App\file1.txt
file03     = PDA\%UID%\file2.txt     > \Application\App\file2.txt
```

the WDp client will replace the variables as follows:

PDA\App\%MODEL%\app.exe	PDA\App\MC9090\app.exe
PDA\Cfg\%HOSTNAME%.cfg	PDA\Cfg\PDA001.cfg
PDA\%UID%\file1.txt	PDA\john\file1.txt
PDA\%UID%\file2.txt	PDA\john\file2.txt

WDP Files

A .wdp file contains a list of actions to be executed when the PDA starts. The client searches the PDA flash folder for .wdp files at start-up. Once the files are processed, they are executed one by one.
There are several levels of wdp files:

.wdp, .wdp0, .wdp1, .wdp2, .wdp3, .wdp4 and .wdp5

At **coldboot** time the files processed are:
.wdp, .wdp0, .wdp1, .wdp2, .wdp3, .wdp4 and .wdp5

At **warmboot** time the files processed are:
.wdp2, .wdp3, .wdp4 and .wdp5

The files are executed in ascendant numerical order, this means the first will be wdp and the last one will be wdp5. In case two or more files have the same number (e.g. cfg.wdp2 and app.wdp2) the order is not defined.

It may exist several ACTION-BOOT during wdp files processing. However, only one of them will be executed. The selection of the ACTION-BOOT that will be executed depends on t*ype* and *immediate* parameters and follows these rules:

- If *immediate=yes* the action is executed immediately after processing the wdp file.
- If both COLDBOOT and WARMBOOT are found, COLDBOOT is executed.
- If there are several WARMBOOT the one with greater delay is executed.
- If there are several COLDBOOT the one with greater delay is executed.

ACTION-BOOT should be used carefully because it may make the PDA loop infinitely. Unless you know what you are doing, it is not recommended to use this action on wdp files that are executed at warmboot time (i.e. wdp2, wdp3, wdp4 and wdp5).

The WDp client can be used as an **installer** by using a wdp5 file. If the client finds a wdp5 file it will exit, even if the file is empty or an action fails.

The WDp client creates a log file ("\WdpInstallerLog.txt") where actions are written. This file can be used for debug purposes.

A wdp file contains two sections: MAIN and ACTIONS.

MAIN section:

[main]
sequentialActions = yes | no
autodelete = yes | no
delayBeforeRunning = delay-time in milliseconds

SequentialActions option allows executing the actions as they were written. If this option is not active or not specified the actions will be executed in predetermined order (see ACTION sections).

Autodelete option allows deleting the wdp file once executed all actions.

DelayBeforeRunning option delays *delay-time* milliseconds the executions of the actions.

ACTION sections:

Defines a set of actions to be executed.
Supported actions are:

ACTION-ZIP, ACTION-CPY, ACTION-ATTRIB, ACTION-REG, ACTION-DLL, ACTION-EXE, ACTION-WAIT-EXE, ACTION-REGEXP, ACTION-DEL, ACTION-REMOVE, ACTION-KILL, ACTION-DELAY and ACTION-BOOT.

See **Actions Reference** for a detailed description of each action.

There are two modes for processing actions: non-sequential and sequential.

In non-sequential mode the actions are processed in the following order:

1) ACTION-ZIP
2) ACTION-CPY
3) ACTION-ATTRIB
4) ACTION-REG
5) ACTION-DLL
6) ACTION-WAIT-EXE
7) ACTION-EXE
8) ACTION-REGEXP
9) ACTION-DEL
10) ACTION-REMOVE
11) ACTION-KILL
13)ACTION-DELAY

14) ACTION-BOOT

To use this mode, deactivate *sequentialActions* in MAIN section.

In sequential mode actions are processed as they were written, except for ACTION-BOOT, which is executed always at the end, no matter its position. In order to use this mode you have to activate *sequentialActions* option in MAIN section.

Actions reference

This is the full list of actions:

ACTION-ZIP
ACTION-CPY
ACTION-ATTRIB
ACTION-REG
ACTION-DLL
ACTION-EXE
ACTION-WAIT-EXE
ACTION-REGEXP
ACTION-DEL
ACTION-REMOVE
ACTION-KILL
ACTION-DELAY
ACTION-BOOT

Almost all actions –except for ACTION-BOOT and ACTION-DELAY (see further down)– have the same structure, which is:

```
[action-name]
stopOnError = yes | no
filecount   = number of files (NN+1)
file00      = file and/or parameters
…
fileNN      = file and/or parameters
```

StopOnError option:

This option allows controlling the execution of the actions. If *stopOnError* is set and the action fails, subsequent actions won't be executed. It is useful when some actions depend on the success of the previous ones.

ACTION-ZIP

Description:

Uncompress a file. Once uncompressed the file is not deleted.

Syntax:

fileXX = filename

If the path or filename contains space characters, use quotation marks. Arguments are optional and they don't require quotation marks.

Example:

```
[action-zip]
filecount =2
file00 = \flash\application.zip
file01 = \configuration.zip
```

ACTION-CPY

Description:

Applies a .cpy-like file. See the appendix "writing cpy files" for a detailed description.

Syntax:

fileXX = filename

Example:

```
[action-cpy]
filecount = 1
file00 = \flash\persist.copy
```

ACTION-ATTRIB

Description:

Changes file/s attribute/s. This action accepts wildcards (“?” and “*”).

Syntax:

fileXX = filename [-r | +r] [-h | +h]

If the path or filename contains space characters, use quotation marks. Arguments are mandatory and they don’t require quotation marks.

Parameters:

+r/-r: sets/unsets “read-only” attribute
+h/-h: sets/unsets “hidden” attribute

Example:

```
[action-attrib]
filecount = 4
file00 = \flash\myapp\file.txt –r
file01 = \flash\*.cfg +r
file02 = \myapp.exe –r –h
file03 = “\flash\program files” +r +h
```

ACTION-REG

Description:

Imports registry keys from a file.

Syntax:

fileXX = filename

Quotation marks are not required for paths or filenames containing space characters.

Example:

```
[action-reg]
filecount = 3
file00 = \flash\wifi.reg
file01 = \app.reg
file02 = \program files\sample.reg
```

The reg file must be compliant with REGEDIT4 format. A keyword '%DELETE%' allows to delete a value or a key.

```
REGEDIT4

[HKEY_CURRENT_USER\Cz\test]
"AllowChange"=dword:00000001

[HKEY_CURRENT_USER\Cz\test]
"AllowChange"=%DELETE%

[HKEY_CURRENT_USER\Cz\test]
%DELETE%
```

ACTION-DLL

Not supported

ACTION-EXE

Description:

Executes an executable file or a document. In order to execute a document file, the file extension must be associated to an application.

Once executed it continues with the subsequent actions.

Syntax:

fileXX = filename [arguments]

If the path or filename contains space characters, use quotation marks. Arguments are optional and they don't require quotation marks.

Example:

```
[action-exe]
filecount = 1
file00 = \windows\app.exe

[action-exe]
filecount = 1
file00 = \windows\MyPage.htm

[action-exe]
filecount = 1
file00 = \windows\MyLink.lnk

[action-exe]
filecount = 1
file00 = "\Program Files\app.exe" -d MyArguments
```

ACTION-WAIT-EXE

Description:

Executes an executable file or a document. In order to execute a document file, the file extension must be associated to an application.

Once executed it waits until the application exits before continuing with the subsequent actions.

Syntax:

fileXX = filename [arguments]

If the path or filename contains space characters, use quotation marks. Arguments are optional and they don't require quotation marks.

Example:

```
[action-wait-exe]
filecount = 1
file00 = \windows\app.exe
file01 = "\Program Files\app.exe" -d MyArguments
```

ACTION-REGEXP

Description:

Exports registry key and its subkeys to a file.

Syntax:

fileXX = file [-a] reg-key

If the path or file contains space characters, use quotation marks.

Abbreviations for main keys:
- HKLM : HKEY_LOCAL_MACHINE
- HKCU : HKEY_CURRENT_USER
- HKCR : HKEY_CLASSES_ROOT
- HKU : HKEY_USERS

To export a key and its subkeys use square brackets in the following way: [HKML\Comm]

It is not possible to export a single-value from a key.

Parameters:

-a: append the content of the registry key at the end the file.

Example:

```
[action-regexp]
filecount = 4
file00 = \myapp.reg [HKCU\software\myapp]
file01 = \devices.reg [HKLM\comm\device1]
file02 = \devices.reg -a [HKLM\comm\device2]
file03 = "\Program Files\devices.reg" -a [HKLM\my key\device2]
```

ACTION-DEL

Description:

Deletes a file. This action accepts wildcards (“?” and “*”).

Syntax:

fileXX = filename

Quotation marks are not required for paths or filenames containing space characters.

Example:

```
[action-del]
filecount = 3
file00 = \myapp.reg
file01 = \flash\1.txt
file02 = \doc files\*.txt
```

ACTION-REMOVE

Description:

Removes directories and files. This action accepts wildcards (“?” and “*”).

Syntax:

fileXX = directory

Example:

```
# remove a directory
[action-remove]
filecount = 1
file00 = \flash\test

# remove only the content of the directory
[action-remove]
filecount = 1
file00 = \flash\test\*.*
```

ACTION-KILL

Description:

Kills a running process.

Syntax:

fileXX = file

The parameter "file" can be either an executable filename or a complete path to an executable file. In case a filename is specified, the client will search the running processes for the first process name that match the filename.

Specify a filename if you are sure that it is the only process with that name running on the system.

Example:

```
[action-kill]
filecount = 3
file00 = myapp.exe
file01= \windows\myapp.exe
file02= \program files\myapp.exe
```

ACTION-DELAY

Description:

Performs a delay.

Syntax:

[action-delay]
delay = delay-time in milliseconds

Example:

```
# delay 3 seconds
[action-delay]
delay = 3000
```

ACTION-BOOT

Description:

Performs a wamboot or coldboot. This action can appear only once in a file and it is always the last action to be executed.

Syntax:

```
[action-boot]
type = warmboot | coldboot
immediate = yes | no
delay = delay-time in seconds
```

Parameters:

immediate: this option is valid only for .wdp files (see **WDP Files**). If this option is set to "yes", ACTION-BOOT is executed immediately when the wdp file processing ends.

delay: wait X seconds before booting.

Examples:

```
# performs a warmboot
[action-boot]
type = warmboot
immediate = no

# performs a coldboot
[action-boot]
type = coldboot
immediate = yes
```

Environment Variables

There is a set of environment variables that can be used in some Actions and CPY files.

System Folders

These variables represent system folders of Windows CE and Windows Mobile and their value depends on the language of the operating system.

Actions that require a path or file name as a parameter are able to use one of these variables.

A table for each operating system with all variables and their corresponding value is showed below. The table only shows values for English operating systems, however these values may change on other languages.

Windows Mobile – English OS:

%PDA_START_MENU%	\Windows\Start Menu
%PDA_PROGRAMS%	\Windows\Start Menu\Programs
%PDA_PERSONAL%	\My Documents
%PDA_STARTUP%	\Windows\Startup
%PDA_FONTS%	\Windows\Fonts
%PDA_FAVORITES%	\Windows\Favorites
%PDA_WINDOWS%	\Windows
%PDA_PROGRAM_FILES%	\Program Files

Windows CE – English OS:

%PDA_START_MENU%	\Windows\Start Menu
%PDA_PROGRAMS%	\Windows\Programs
%PDA_PERSONAL%	\My Documents
%PDA_STARTUP%	\Windows\Startup
%PDA_FONTS%	\Windows\Fonts
%PDA_FAVORITES%	\Windows\Favorites
%PDA_WINDOWS%	\Windows
%PDA_DESKTOP%	\Windows\Desktop
%PDA_PROGRAM_FILES%	\Program Files

Example:

```
# execute an application
[action-exe]
filecount = 1
file00 = %PDA_PROGRAM_FILES%\MyApplication\app.exe
```

Appendix A: Writing CPY files

A CPY specifies a list of files to copy.

Syntax:

source > target

source: complete path to the source file
target: complete path to the target file

Quotation marks are not required for source and target.
Environment variables are allowed on source and target.

A CPY file example:

```
\wdp\flexvnc.exe            >     \Windows\flexvnc.exe
\wdp\flexftpserver.exe      >     \Windows\flexftpserver.exe
\wdp\flexrpvserver.exe      >     \Windows\flexrpvserver.exe
\flash\myapp.lnk            >     \program files\myapp.lnk


# using environment variables
\wdp\flexvnc.exe            >     %PDA_WINDOWS%\flexvnc.exe
\wdp\flexftpserver.exe      >     %PDA_WINDOWS%\flexftpserver.exe
\wdp\flexrpvserver.exe      >     %PDA_WINDOWS%\flexrpvserver.exe
\flash\myapp.lnk            >     %PDA_PROGRAM_FILES%\myapp.lnk
%PDA_PROGRAM_FILES%\a.lnk   >     %PDA_PROGRAMS%\a.lnk
```