asustor

App Central: Developer's Guide

For APKG 2.0

Revision: 2.7.0
Update: February 6, 2017

Table of Content

1 System Requirements

1.1 Build Machine

1. Ubuntu 14.04 (recommended), 64-bit
2. Other OS (Windows, Mac OS, etc.)
3. asustor cross compiler toolchain (Intel 64bit, Intel 32bit, ARM 32bit)
4. Bourne Shell
5. Python 2.7
6. Text editor (vim, gedit, eclipse, etc.)

1.2 Target Machine

Product Series	CPU	Arch
AS70	Intel Core i3 4330 Intel Xeon E3	Intel x86-64
AS63/64	Intel Celeron ()	Intel x86-64
AS61/62/31/32	Intel Celeron (Braswell)	Intel x86-64
AS-6XX	Intel Atom (Cedarview)	Intel x86-64
AS50/51	Intel Celeron (Bay trail)	Intel x86-64
AS-2XX/3XX	Intel Atom (Evansport)	Intel i386
AS10	Marvell ARMADA	ARM 32-bit

ADM information		ADM 2.5.3	ADM 2.6	ADM 2.7
Shell			BusyBox v1.19.3	
LAMP			Apache 2.2.31 MariaDB 10.0.28 PHP 5.6.14	
Linux Kernel	**AS10**	3.10.70	3.10.70[1]	3.10.70
	AS2/2TE/3	3.4.25	3.4.25	3.4.25
	AS63/64	N/A	N/A	4.4.24
	AS31/61/62	4.1.0	4.1.0	4.4.24
	AS50/51	3.12.20	4.1.0	4.4.24
	AS6	3.12.20	3.12.20	3.12.20
	AS70	3.12.20	4.1.0	4.4.24
The GNU C Library		eglibc 2.13	glibc 2.22	

[1] Supports hardware floating point

App Central system path	/usr/local/[2]
App repository	/usr/local/AppCentral/[3]
App home	/usr/local/AppCentral/$APP_NAME

[2] Default system folder: /usr/local/{/usr/local/{bin,etc,lib,lib64,sbin,tmp}

[3] Files should only be copied to the $APP directory. If you need to place files in other locations, you may use a soft link and link the files under /usr/local/.

2 About APKG

APKG is a package management system for developing and managing ASUSTOR NAS apps. Different APKG versions may have different APK (ASUSTOR application package file) formats, therefore, using the correct build tool/script is very important before building any apps.

2.1 App Central Guidelines

We review all Apps to ensure they are reliable, perform as expected, and are free of offensive material.
Before submitting your new or updated Apps for review, please ensure your Apps comply with these guidelines:

1. Ensure the network port(s) used by the application is not taken.
   What network ports are used by ASUSTOR services?
2. If the application creates working folders and files, please set the owner as "admin", group as "administrator" and access permission as "766".
3. If the application contains functions that can browse or select directories, please only allow the user to choose directories from the data volumes. (volume1~n)

2.2 Getting Started

Before building your own apps, there is one thing you must know – config.json. It is the fundamental of each app which contains the necessary information about your app and the required environment for installation. The configurations have been divided into three categories (app, desktop and install) in config.json, and we will explain them in the next three sections respectively.

Please see section 4.3 Best Practice for the examples of config.json.

Name	Description
general	This section contains the information about the package and will be displayed in App Central.
adm-desktop	Optional. This section defines the type of this app.
register	Optional. This section is used for installation.

2.3 config.json: general

This section defines the basic information of this app.

Key	Description	Type	Note
general	App information section.	Object	
package	This is the package name. It is used to distinguish between different apps. It must be a unique name.	String	
name	The app name which will be displayed in App Central.	String	
version	The version of this package.	String	
depends	The dependent package list of this package. Before a package is being installed or upgraded, these packages must be installed first.	Array(String)	1. A 2. B (>= 1.0) 3. C (<= 2.0) 4. D (= 1.5) 5. E (>= 1.0, <= 2.0)
conflicts[4]	(RESERVED) Not used.	Array(String)	
developer	Name of developer.	String	
maintainer	Name of maintainer.	String	
email	Your email address.	String	
website	Your website or any associated links.	String	
architecture[5]	This is used to identify the platform.	String	x86-64 / i386/ arm/ any
firmware	Required firmware version.	String	2.0
model	This is used to identify the NAS models.	String	10xx; 31xx; 50xx/51xx; 61xx/62xx; 63xx/64xx; 70xx
default-lang	To specific the default language of license agreement which will be shown in the installation process.	String	cs / da/ de/ en-US / es /fi /fr-FR/ hu/ it-IT/ ja-JP/ ko-KR/ nl-NL/ no/ pl/ pt/ ru-RU/ sv/ tr/ zh-TW/ zh-CN
memory-limit	Minimum memory size requirement. The app will be disabled when the installed memory is lower than this size.	Integer(MB)	
memory-advice	Memory size advice.	Integer(MB)	

Note:

1. All words are case sensitive.

[4] Leave the field empty.

[5] If your App is common web applications (php, html), use 'any' for architecture.

2.4 config.json: adm-desktop

This section is used to define the type of this app and its default privileges. There are two major objects in this section: icon & privilege

Key	Description	Type	Note
adm-desktop	Icon and privilege settings section.	Object	
app[6]	ADM desktop icon settings.	Object	
privilege[7]	App privilege settings.	Object	

2.4.1 app

This object defines the type of the app and currently there are four types of apps. They are: ***internal***, ***external***, ***webserver***, and ***custom***. Please note that both "***internal***" & "***external***" are reserved for ASUSTOR in-house development. Most 3rd party developers will use "***webserver***" & "***custom***".

2.4.1.1 Type: Web Apps

Web Apps are for common web applications such as **phpMyAdmin** and **WordPress**. It runs on the system built-in Apache web server. This will potentially be used by 3rd party maintainers or developers. Here is an example of this kind of App:

```
"app":{
     "type":"webserver"
},
```

Key	Description	Type	Note
app	ADM desktop icon settings	Object	
type	The type for this app	String	
session-id	Send current session id when launching application	String	true / false

[6] There are four types so far, they are: internal, external, webserver, customize. Each type has its own format.
[7] This is used to define the default permission of this app.

2.4.1.2 Type: Custom Apps

Most 3rd party developer will use this. You can run your own web server, define the protocol, port and URL. Here is an example of this kind of App:

```
"app":{
    "type":"custom",
    "protocol":"http",
    "port": 39876,
    "url": "/"
}
```

Key	Description	Type	Note
app	ADM desktop icon settings	Object	
type	The type for this app	String	
protocol	The network protocol	String	http / https
port	The port number	Integer	
url	The URL of your web page	String	

2.4.2 privilege

Key	Description	Type	Note
accessible	You can define the group(s) which will be able to use this app by default. [users] represents all system users while [administrators] represents the specific user(s) who have the administration rights.	String	
customizable	This determines if the access rights to this app can be modified in [Access Control] -> [App Privilege]. For example, if “accessible” is set to “administrators”, but you would like to allow another non-administrator user to access the app, then you should use “true” here.	Boolean	

Note:

1. All words are case sensitive.

2. “privilege” will not be able to restrict access to web applications since most of them have their own account system. It only can be used to determine whether the desktop icon is visible or invisible to users.

2.5 config.json: register

```
"register":{
        "symbolic-link":{
        },
        "share-folder":[
            {
                "name":"Download",
                "description":"Download default shared folder"
            }
        ],
        "port":[
            "9999",
            "55555"
        ],
        "boot-priority":{
            "start-order":20,
            "stop-order":80
        },
        "prerequisites":{
            "enable-service":[],
            "restart-service":[]
        }
    }
```

Key	**Description**	**Type**	**Note**
register	Install settings section	Object	
symbolic-link	The link used for create soft link to /usr/local folder in this App.	Object	/bin, /etc, /lib, /lib64, /sbin, /var
share-folder	This is where you can define the default directories (shared folders) for this app. These directories will be	Array(Object)	

	created automatically while installing this app, and if the specified directory already exists, the app will ignore this and just use the directory.		
name	The name of this share.	String	
description	The description of this share.	String	
port	Port numbers of service used.	Array(Integer)	
boot-priority	Priority of service start-stop.	Object	
start-order	Service start with script: S{$PRIORITY}{$APP_NAME}.	Integer	00 ~ 99
stop-order	Service stop with script: K{$PRIORITY}{$APP_NAME}.	Integer	00 ~ 99
prerequisites	After the package was installed or upgraded, these services must start or restart.		1. samba 2. afp 3. nfs 4. ftp 5. webdav 6. httpd 7. mysql
enable-service	These services must be started or enabled.	Array(String)	
restart-service	These services must be restarted.	Array(String)	

Note:

1. All words are case sensitive.
2. All keys of above are optional.

3 Building Your App

In this chapter, we will introduce the package source structure, utilities for building apps, and final app structure.

3.1 *Prepare Your Package Source*

Your package source should contain at least one folder – CONTROL. You can also add self-defined folders to store other files, such as www, lib, etc.

Folder Name	Description
CONTROL[8]	This folder is used to store some necessary files, such as config.json, icons and scripts. Please refer to 3.1.1 CONTROL for more details.
www	Optional. This folder is for common web applications (php, html) which need to be run on a web server such as phpMyAdmin and Joomla!. (Apache comes with ADM and can be found under [Services] -> [Web Server])This is where the source files will be placed.
(OTHERS) [9]	Self-defined folders, such as bin, etc, lib, lib64, etc. You are free to define any new folders here.

Note:

1. The CONTROL folder name is case sensitive.

3.1.1 CONTROL

This folder is used to store app information, configuration, icons and other hook scripts. Please see the chart below.

File Name	Description	File Type
config.json	This file contains the information displayed in App Central and setting environment in the installation process.	JSON file
icon.png	90 x 90 pixels in PNG format, which is shown in App Central and used for ADM[10] desktop Icon.	PNG image, transparent
description.txt	The general description of the app.	Text file
changelog.txt	The change log of this revision.	Text file

[8] This folder is used to store app information, configuration, icons and other hook scripts.
[9] There are also other default folders such as bin, etc, lib, lib64, etc. You are free to define any new folders here.
[10] ASUSTOR Data Master, Web Desktop UI.

license	Optional. This **folder** contains the license agreement files which will be shown in the installation process.	Text file
pre-install.sh	Optional. This hook script which is executed before installation.	Bourne shell script
pre-uninstall.sh	Optional. This hook script which is executed before uninstalling / upgrading.	Bourne shell script
post-install.sh	Optional. This hook script which is executed after installation / upgrading.	Bourne shell script
post-uninstall.sh	Optional. This hook script which is executed after uninstalling.	Bourne shell script
start-stop.sh	Optional. The init.d script to start and stop an app. This script is for daemon App, it will be executed automatically after booting or before power off. Ex: . /etc/script/lib/apkg_path.sh	Bourne shell script

Note:

1. All file names are case sensitive and fixed.
2. These files are necessary:
   a. config.json
3. The hook scripts will be executed if available, or you can just ignore this.

3.2 Utilities for Building Apps

Here is the Linux script which can help you to build your own app.

3.2.1 Packing an App

Usage:

```
Apkg-tool.py create <pkg_directory> [<destination_directory>]
```

Example:

```
root@build-machine:/as_build/apk# apkg-tool.py create download-center
```

3.3 *Final App Structure*

After executing the apkg-tool.py script, all source folders and files (as in 3.1 Prepare Your Package Source) will be compressed, and an app with the name ***PACKAGE_VERSION_ARCHITECTURE.apk*** will be generated automatically.

To check your app structure, you can decompress the apk file with uzip. You should then be able to see the following files in the apk.

File Name	Description	File Type	Mandatory
apkg-version	This file specifies the version of the apk format.	Text file	Yes
control.tar.gz	This is a compressed file in .tar.gz format containing all the files that are required for configuration and display, such as configuration files, icons, license, daemon control file, or hook scripts.	Gzipped tarball	Yes
data.tar.gz	This is a compressed file in .tar.gz format containing all source files, such as executable binary, library, or UI files.	Gzipped tarball	Yes

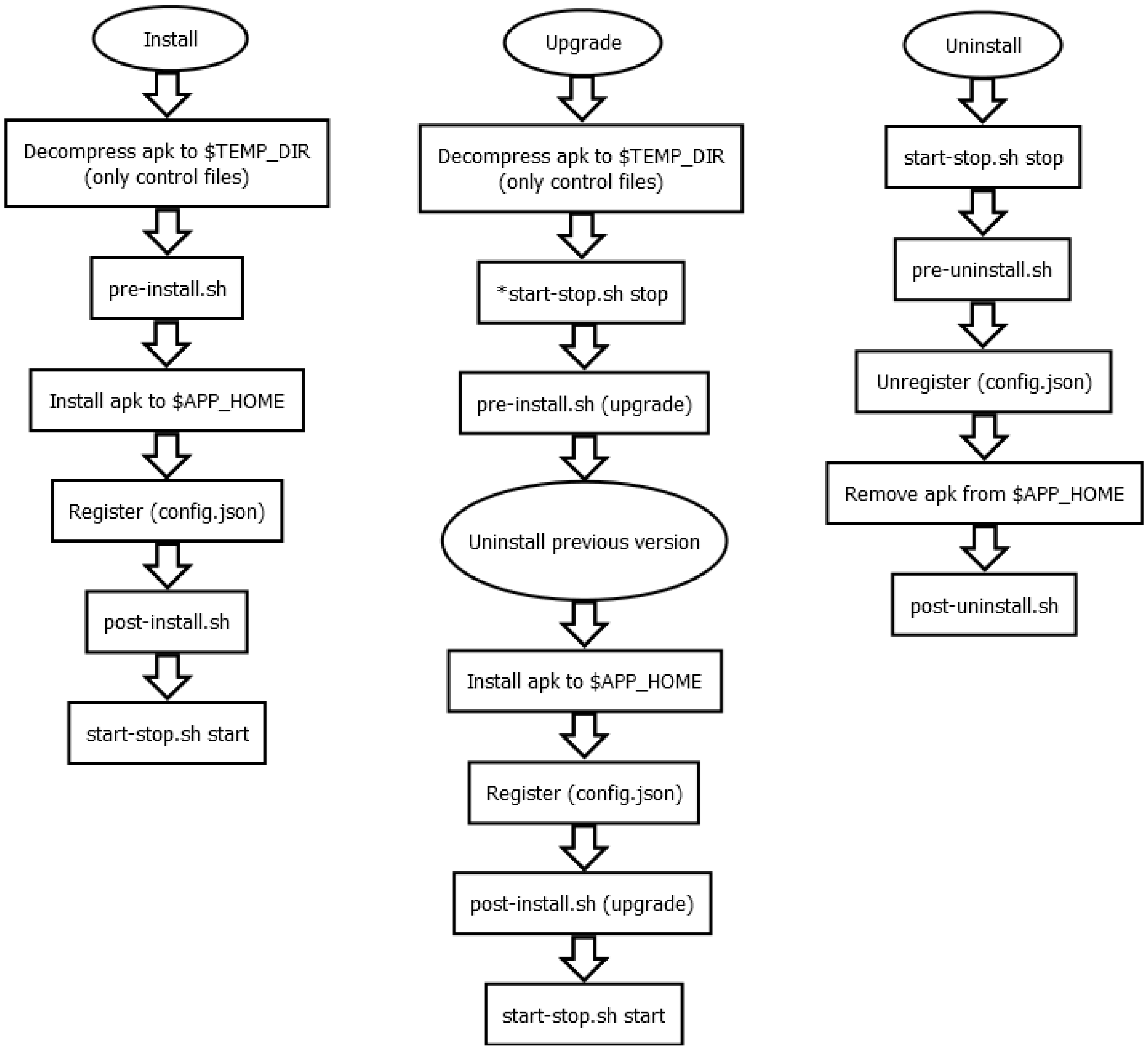
Note:

1. apkg-version file contents[11]:
   2.0
2. Package name format:
   PACKAGE_VERSION_ARCHITECTURE.apk

[11] Apkg version 2.0 is currently in use.

4 Appendix

4.1 APKG State Transition Diagram



*comes from the currently installed package.

4.2 Script Environment Variables

Several variables are exported by App Central and can be used in the scripts. Descriptions of these variables are given below:

Variable Name	Description
AS_NAS_ARCH	The type of CPU architecture.
AS_NAS_KERNEL	The version of NAS kernel.
AS_NAS_MODEL	The name of NAS model.
AS_NAS_FIRMWARE	The version of NAS firmware.
AS_NAS_HOSTNAME	The hostname of NAS.
AS_NAS_TIMEZONE	The time zone setting of NAS.
AS_NAS_INET4_ADDR_0	The IPv4 address of first network interface card.
AS_NAS_INET4_ADDR_1	The IPv4 address of second network interface card.
APKG_BASE_DIR	App Central system root.
APKG_REPO_DIR	Apps repository directory.
APKG_PKG_NAME	The package name of App which is defined in config.json.
APKG_PKG_VER	The version of App which is defined in config.json.
APKG_PKG_INST_VER	The version of App which has been installed on App Central.
APKG_PKG_DIR	App directory in which the package is stored.
APKG_PKG_STATUS	Package status can be represented by these values: install, upgrade, uninstall.
APKG_TEMP_DIR	App Central randomly generates a dir name for a script to store the configuration.

Note:

1. All words are case sensitive.

4.3 Best Practice – LooksGood

LooksGood is a in house party application which allows user to directly stream videos from NAS to Web browser and mobile devices. Below are the samples of its config.json, source layout and package layout.

4.3.1 App Configuration – config.json

```
{
    "general":{
        "package":"looksgood",
        "name":"LooksGood",
        "version":"1.0.0.r2016",
        "depends":[
            "python(>=2.7.3.r13)",
            "xorg(>=10.14.6.377)"
        ],
        "conflicts":[],
        "developer":"ASUSTOR",
        "maintainer":"ASUSTOR",
        "email":"support@asustor.com",
        "website":"http://www.asustor.com/",
        "architecture":"x86-64",
        "firmware":"2.4.0",
        "model":[
            "50xx",
            "51xx"
        ]
    },
    "adm-desktop":{
         "app":{
              "type":"webserver",
              "session-id":true
         },
         "privilege":{
              "accessible":"users",
              "customizable":true
         }
    },
    "register":{
        "share-folder":[
            {
              "name":"Video",
              "description":"Media Station default shared folder"
            }
        ],
        "prerequisites":{
              "enable-service":["httpd"],
              "restart-service":[]
        },
        "boot-priority":{
            "start-order":95,
            "stop-order":5
        },
        "port":[9900, 9901, 9902, 9903, 9904]
    }
}
```

4.3.2 Package Source Layout

Below is the package layout of Download Center. You can use the attached **apkg-tool.py** Linux script to build your own app automatically.

```
root@build-machine:/as_build/apk/download-center# tree
├─── bin
│    └─── dlcd
├─── CONTROL
│    ├─── config.json
│    ├─── icon.png
│    ├─── license.txt
│    ├─── decription.txt
│    ├─── changlog.txt
│    ├─── post-install.sh
│    ├─── post-uninstall.sh
│    ├─── pre-install.sh
│    ├─── pre-uninstall.sh
│    └─── start-stop.sh
├─── etc
│    ├─── plugin
│    │    └─── rss.json
│    │    └─── search.json
│    └─── settings.json
├─── lib
│    ├─── libdlcenter.so -> libdlcenter.so.0.0
│    ├─── libdlcenter.so.0 -> libdlcenter.so.0.0
│    ├─── libdlcenter.so.0.0
│    ├─── libevent-2.0.so.5 -> libevent-2.0.so.5.1.1
│    └─── libevent-2.0.so.5.1.1
└─── webman
     ├─── dlcenter.cgi
     ├─── downloadCenter.js
     ├─── images
     │         ├─── icon-app-task.png
│          ├───── icon.png
│          ├───── icon-title.png
│          ├───── left_icon1.png
│          └───── left_icon2.png
└───── langs
     └───── lang-en-US.js
```