

QDK Quick Start Guide

Tutorial to Build Your Own QPKG

Agenda

- What is QDK
- Download QDK
- Install QDK
- Build Your Own QPKG
 - Generate environment for QPKG
 - Configure QPKG
 - Customize QPKG routines
 - Add files to QPKG
 - Generate QPKG file
- Learn More

What is QDK

- QDK is used to build QPKG files/applications for QNAP Turbo NAS.
- QDK started out as a simple modification of the first official release of the QPKG SDK, but now supersedes it.
- License: GPL

Download QDK

- To download QDK:
 - http://wiki.qnap.com/wiki/QPKG_Development_Guidelines#Downloads
 - http://files.qnap.com/download/Storage/QPKG/QDK_2.0.1.zip

Install QDK

- Install “QDK_2.0.1.qpkg” in NAS UI



Install QDK

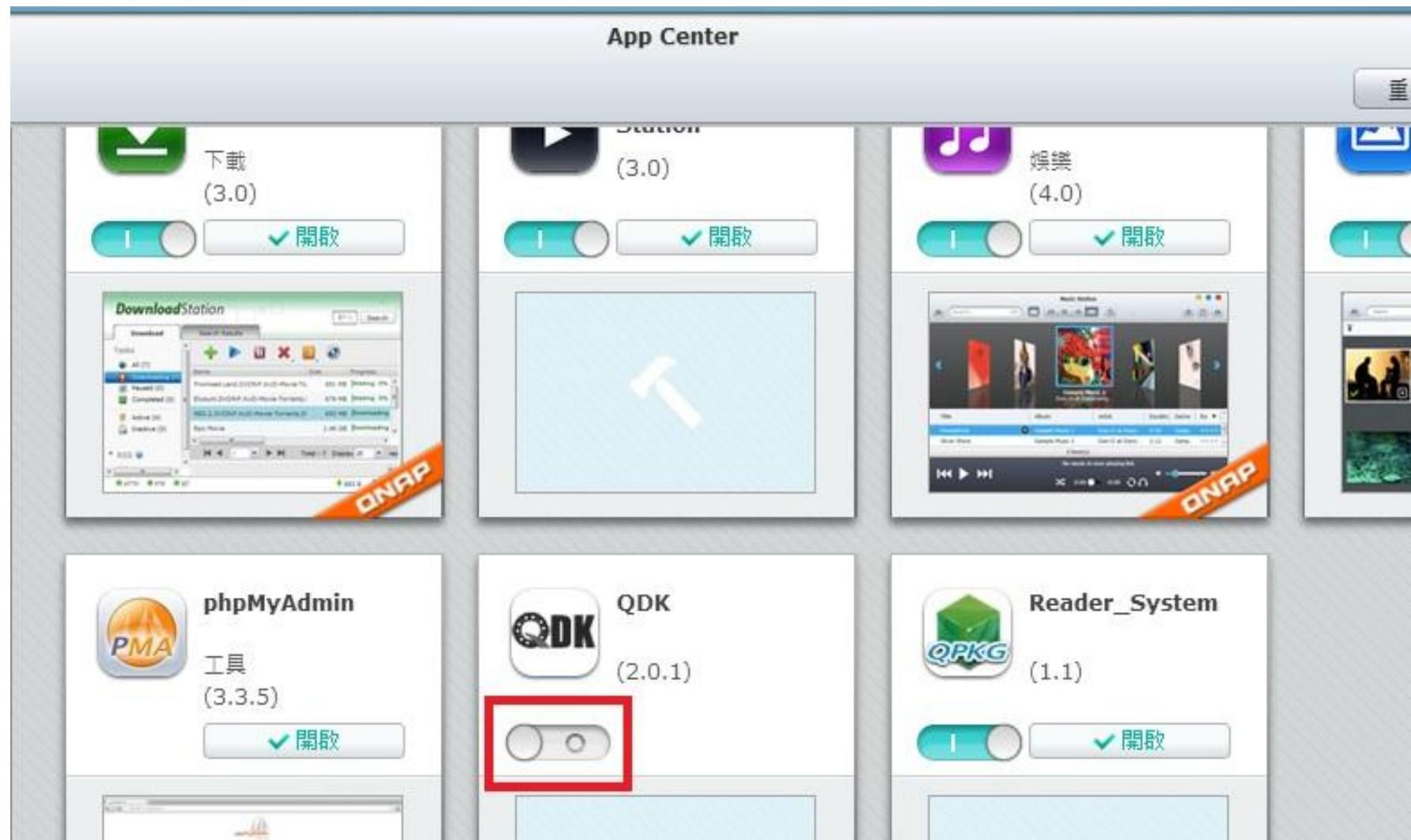


Install QDK



Enable QDK

- Enable QDK in NAS UI



Build Your Own QPKG (1/5)

- Generate environment for QPKG
 - Use SSH client to connect to your NAS
 - Issue below commands to create an environment of your own QPKG (assuming to-be-built QPKG name is “MyQPKG”)

```
[~] ln -s /bin/sh /bin/bash  
[~] cd `getcfg QDK Install_Path -f /etc/config/qpkg.conf  
[/share/HDA_DATA/.qpkg/QDK] # qbuild --create-env MyQPKG
```
 - A folder named “MyQPKG” is then generated

```
[/share/HDA_DATA/.qpkg/QDK] # ls  
MyQPKG/ bin/ qdk* scripts/ template/  
[/share/HDA_DATA/.qpkg/QDK] # cd MyQPKG/  
[/share/HDA_DATA/.qpkg/QDK/MyQPKG] # ls  
arm-x09/ build/ icons/ qpkg.cfg x86/  
arm-x19/ config/ package_routines shared/ x86_64/
```

Screenshot

```
[~] # ln -s /bin/sh /bin/bash
[~] # cd `getcfg QDK Install_Path -f /etc/config/qpkg.conf`
[/share/HDB_DATA/.qpkg/QDK] # qbuild --create-env MyQPKG
[/share/HDB_DATA/.qpkg/QDK] # ls
MyQPKG/  bin/      qdk*      scripts/  template/
[/share/HDB_DATA/.qpkg/QDK] # cd MyQPKG
[/share/HDB_DATA/.qpkg/QDK/MyQPKG] # ls
arm-x09/          config/          package_routines  shared/          x86_64/
arm-x19/          icons/          qpkg.cfg           x86/
```

Build Your Own QPKG (2/5)

- Configure QPKG

- Edit the content of qpkg.cfg
 - QPKG_NAME: Name of the QPKG
 - QPKG_VER: Version of the QPKG
 - QPKG_AUTHOR: Author of the QPKG

```
[/share/HDA_DATA/.qpkg/QDK/MyQPKG] # vi qpkg.cfg
```

```
# Name of the packaged application.  
QPKG_NAME="MyQPKG"  
# Version of the packaged application.  
QPKG_VER="0.1"  
# Author or maintainer of the package  
QPKG_AUTHOR="admin"
```

Screenshot

A screenshot of a terminal window titled "1. ssh". The window contains a configuration file with various parameters for a package named "MyQPKG". The file includes details like version ("0.1"), author ("admin"), license (""), summary (""), and service program ("MyQPKG.sh"). It also specifies required packages ("Python >= 2.7, Optware | opkg, OPT/openssh") and conflict packages ("Python, OPT/sed"). Configuration files are set to "myApp.conf" and "/etc/config/myApp.conf". The service port is left blank (""). The PID file location is also blank (""). The web interface relative path is blank ("") and the port is blank (""). The chroot environment location is specified as "qpkg.cfg". The bottom of the terminal shows the command ":wq" and status indicators [3%] and 2,18.

```
# Name of the packaged application.  
QPKG_NAME="MyQPKG"  
# Version of the packaged application.  
QPKG_VER="0.1"  
# Author or maintainer of the package  
QPKG_AUTHOR="admin"  
# License for the packaged application  
#QPKG_LICENSE=""  
# One-line description of the packaged application  
#QPKG_SUMMARY=""  
  
# Preferred number in start/stop sequence.  
QPKG_RC_NUM="101"  
# Init-script used to control the start and stop of the installed application.  
QPKG_SERVICE_PROGRAM="MyQPKG.sh"  
  
# Specifies any packages required for the current package to operate.  
#QPKG_REQUIRE="Python >= 2.7, Optware | opkg, OPT/openssh"  
# Specifies what packages cannot be installed if the current package  
# is to operate properly.  
#QPKG_CONFLICT="Python, OPT/sed"  
# Name of configuration file (multiple definitions are allowed).  
#QPKG_CONFIG="myApp.conf"  
#QPKG_CONFIG="/etc/config/myApp.conf"  
# Port number used by service program.  
#QPKG_SERVICE_PORT=""  
# Location of file with running service's PID  
#QPKG_SERVICE_PIDFILE=""  
# Relative path to web interface  
#QPKG_WEBUI=""  
# Port number for the web interface.  
#QPKG_WEB_PORT=""  
  
# Location of the chroot environment (only TS-x09)  
qpkg.cfg  
:wq
```

Build Your Own QPKG (3/5)

- Customize QPKG routines
 - Content of file “package_routines”
 - pkg_pre_install() : routines before install
 - pkg_install() : routines during install
 - pkg_post_install() : routines after install
 - PKG_PRE_REMOVE : routines before uninstall
 - PKG_MAIN_REMOVE : routines during uninstall
 - PKG_POST_REMOVE : routines after uninstall
 - Content of file “shared/MyQPKG.sh”
 - Start : routines when starting the QPKG
 - Stop : routines when stopping the QPKG

Screenshot

The screenshot shows a terminal window titled "1.ssh". The window contains a shell script with the following content:

```
#!/bin/sh
CONF=/etc/config/qpkg.conf
QPKG_NAME="MyQPKG"

case "$1" in
    start)
        ENABLED=$(./sbin/getcfg $QPKG_NAME Enable -u -d FALSE -f $CONF)
        if [ "$ENABLED" != "TRUE" ]; then
            echo "$QPKG_NAME is disabled."
            exit 1
        fi
        : ADD START ACTIONS HERE
        ;;

    stop)
        : ADD STOP ACTIONS HERE
        ;;

    restart)
        $0 stop
        $0 start
        ;;

    *)
        echo "Usage: $0 {start|stop|restart}"
        exit 1
esac

exit 0
~
```

The script defines variables `CONF` and `QPKG_NAME`. It then uses a `case` statement to handle three commands: `start`, `stop`, and `restart`. For each command, it checks if the package is enabled using `getcfg`. If disabled, it exits with code 1. Otherwise, it executes actions defined in the script. The `stop` and `restart` cases both call the `stop` and `start` functions respectively. The `*` case handles any other input by printing usage instructions and exiting with code 1. Finally, it exits with code 0.

At the bottom of the terminal window, there is a status bar with the text "shared/MyQPKG.sh" and "shared/MyQPKG.sh" 29L, 441C, and a battery icon indicating 3% power remaining.

Build Your Own QPKG (4/5)

- Add files to QPKG
 - Put files in below folders for different purposes:
 - shared/: Platform-independent files and folders
 - arm-x09/ arm-x19/ x86/ x86_64/: Platform-dependent files and folders
 - icons/: icon files
 - config/: config files

```
[/share/HDA_DATA/.qpkg/QDK/MyQPKG] # ls  
arm-x09/      build/      icons/      qpkg.cfg      x86/  
arm-x19/      config/     package_routines  shared/      x86_64/
```

Screenshot

```
[/share/HDB_DATA/.qpkg/QDK/MyQPKG/shared] # mkdir web  
[/share/HDB_DATA/.qpkg/QDK/MyQPKG/shared] # cd web  
[/share/HDB_DATA/.qpkg/QDK/MyQPKG/shared/web] # vi index.html
```

A screenshot of a terminal window titled "1. ssh". The window shows the following command history and file content:

```
1. ssh
[/share/HDB_DATA/.qpkg/QDK/MyQPKG/shared] # mkdir web
[/share/HDB_DATA/.qpkg/QDK/MyQPKG/shared] # cd web
[/share/HDB_DATA/.qpkg/QDK/MyQPKG/shared/web] # vi index.html
```

The file "index.html" contains the following code:

```
<html>
<title>This is MyQPKG</title>
</html>
```

The terminal window has a status bar at the bottom showing "index.html [+] :wq [66%] 2,29".

Build Your Own QPKG (5/5)

- Generate QPKG file
 - Use below command to build the QPKG file

```
[/share/HDA_DATA/.qpkg/QDK/MyQPKG] # qbuild
Creating archive with data files...
Creating archive with control files...
Creating QPKG package...
```
 - The QPKG file will be generated in the build folder

```
[/][/share/HDA_DATA/.qpkg/QDK/MyQPKG] # cd build/
[/share/HDA_DATA/.qpkg/QDK/MyQPKG/build] # ls
MyQPKG_0.1.qpkg
```

Screenshot

```
[/share/HDB_DATA/.apkg/QDK/MyQPKG/shared/web] # chmod 777 index.html
[/share/HDB_DATA/.apkg/QDK/MyQPKG/shared/web] # cd ..
[/share/HDB_DATA/.apkg/QDK/MyQPKG/shared] # ls
MyQPKG.sh* web/
[/share/HDB_DATA/.apkg/QDK/MyQPKG/shared] # cd ..
[/share/HDB_DATA/.apkg/QDK/MyQPKG] # ls
arm-x09/          config/          package_routines shared/           x86_64/
arm-x19/          icons/          apkgo.cfg        x86/
[/share/HDB_DATA/.apkg/QDK/MyQPKG] # qbuild
Creating archive with data files...
Creating archive with control files...
Creating QPKG package...
[/share/HDB_DATA/.apkg/QDK/MyQPKG] # ls build/
MyQPKG_0.1.apkg
```

Learn More

- Learn more from QDK_2.0.pdf in below URL
 - http://files.qnap.com/download/Storage/QPKG/QDK_2.0_doc.zip