

Synology DiskStation Manager

3rd-Party Apps Developer Guide

Synology[®]

2010-11-30



Synology Inc.
© 2010 Synology Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Synology Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Synology's copyright notice.

The Synology logo is a trademark of Synology Inc.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Synology retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Synology-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Synology is not responsible for typographical errors.

Synology Inc.
6F-2, No. 106, Chang-An W.
Rd. Taipei 103, Taiwan

Synology and the Synology logo are trademarks of Synology Inc., registered in the United States and other countries.

Marvell is registered trademarks of Marvell Semiconductor, Inc. or its subsidiaries in the United States and other countries.

Freescale is registered trademarks of Freescale Semiconductor, Inc. or its subsidiaries in the United

States and other countries.

Other products and company names mentioned herein are trademarks of their respective holders.

Even though Synology has reviewed this document, SYNOLOGY MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY. IN NO EVENT WILL SYNOLOGY BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Synology dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Table of Content

Introduction to the Synology DiskStation Manager 3rd-Party Apps Developer Guide	4
Who Should Read This Document?	4
System Requirement	4
Compiling an Application	5
Downloading DSM Tool Chain.....	6
Compiling	7
Compiling open source projects	8
Compiling kernel module	10
Installation.....	12
Put the application in /usr/local	12
Run the application when the system boots	12
Integrating into the Synology DiskStation Manager 2.3.....	14
Startup.....	14
Application.cfg.....	15
Example	16
Adding You Application as Desktop Item.....	18
Desktop.cfg	19
Integrating into the Synology DiskStation Manager 3.0.....	21
Startup.....	21
config.....	22
Integrating with Synology web authentication.....	24

Introduction to the Synology DiskStation Manager 3rd-Party Apps Developer Guide

The Synology DiskStation (DiskStation series, Cube Station series, and Rack Station series) is developed on Linux kernel. Due to frequent requests from Synology users and system integrators to install 3rd-party applications on our products, Synology has prepared this development guide to help you to:

1. Compile applications to be run on the Synology DiskStation
2. Integrate the applications into DiskStation's operation system, a.k.a. Synology DiskStation Manager (DSM)
3. Locate applications to the right path to keep them intact after DSM upgrade
4. Integrate the applications into Synology web authentication interface

Who Should Read This Document?

This document is for Synology users and system integrators interested in adding their applications to their Synology DiskStations.

This document assumes you have a basic understanding of the Linux environment. Many resources exist in print and on the web for learning about Linux application development. If you are new to this, you may want to learn some basics before starting to use this document.

System Requirement

The Synology DiskStation must be installed with DSM 2.0-0636 or above.

Compiling an Application

The Synology DiskStation uses an embedded SoC as the CPU, which has a different architecture from x86-based PCs. There are two platforms, ARM and PowerPC, for different Synology DiskStation models. In order to run 3rd-party applications on the Synology DiskStation, it is necessary to compile the applications into an executable format for the corresponding platform.

The table below lists the CPU, architecture, Endianness, and Linux kernel version of each Synology DiskStation model. The information will help you to determine the correct way to proceed.

Model	CPU	Arch	Endianness	Linux
CS407, DS107+, DS207+, RS407	Marvell 5281	ARM	Little Endian	2.6.15
DS109, DS209, DS409, DS409slim, DS509, RS409	Marvell 6281	ARM	Little Endian	2.6.24
DS110j, DS210j, DS410j, DS211j, DS411j	Marvell 6281	ARM	Little Endian	2.6.32
DS111, DS211	Marvell 6282	ARM	Little Endian	2.6.32
CS407e, DS207, DS209j	Freescale 8241	PowerPC	Big Endian	2.6.24
DS107e, DS107, DS108j, DS109j	Freescale 8241	PowerPC	Big Endian	2.4
DS508, RS408, RS408-RP	Freescale 8543	PowerPC	Big Endian	2.6.32
DS109+, DS209+, DS409+, RS409+, RS409RP+	Freescale 8533	PowerPC	Big Endian	2.6.32
DS110+, DS210+, DS410	Freescale 8533 E	PowerPC	Big Endian	2.6.32
DS710+	Intel Atom D410	Intel x86	Little Endian	2.6.32
DS1010+, DS411+, RS810+, RS810RP+	Intel Atom D510	Intel x86	Little Endian	2.6.32

To compile an application for the Synology DiskStation, a compiler that runs on a Linux PC is needed in order to generate an executable file for the Synology DiskStation. The process is called “cross compiling”, and the set of compiling tools (compiler, linker, etc) is called “tool chain”.

Downloading DSM Tool Chain

To download DSM tool chain, please go to <http://sourceforge.net/projects/dsgpl/files>. The table below shows the tool chain file name for each Synology DiskStation model:

Model	Tool chain
CS407, DS107+, DS207+, RS407	gcc343_glibc232_88f5281.tgz
DS109, DS209, DS409, DS409slim, DS509, RS409	gcc421_glibc25_88f628x.tgz
DS110j, DS210j, DS410j, DS211j, DS411j	gcc421_glibc25_88f628x.tgz
DS111, DS211	gcc421_glibc25_88f628x.tgz
CS407e, DS207, DS209j	gcc334_glibc233_ppc_2624.tgz
DS107e, DS107, DS108j, DS109j	gcc334_glibc233_ppc_2422.tgz
DS508, RS408, RS408-RP	gcc343_glibc234_854x.tgz
DS109+, DS209+, DS409+, RS409+, RS409RP+	gcc343_glibc234_853x.tgz
DS110+, DS210+, DS410	gcc343_glibc234_853x.tgz
DS710+, DS1010+, DS411+, RS810+, RS810RP+	gcc421_glibc236_x86.tgz

After you've downloaded the DSM tool chain, un-tar it into **/usr/local/** using the following command:

```
# tar xzpf gcc343_glibc232_88f5281.tgz -C /usr/local/
```

Please make sure the tool chain is put into the **/usr/local** directory to ensure proper integration.

Compiling

Now you can start to compile the application. For example, if you have an application called "sysinfo.c", with the content below:

```
#include <sys/sysinfo.h>

int main()
{
    struct sysinfo info;
    int ret;

    ret = sysinfo(&info);
    if (ret != 0) {
        printf("Failed to get system information.\n");
        return -1;
    }
    printf("Total RAM: %u\n", info.totalram);
    printf("Free RAM: %u\n", info.freeram);
    return 0;
}
```

To compile, run the command:

```
# /usr/local/arm-marvell-linux-gnu/bin/arm-marvell-linux-gnu-gcc
sysinfo.c -o sysinfo
```

You can also write a Makefile for it:

```
EXEC= sysinfo
OBJS= sysinfo.o

CC= /usr/local/arm-marvell-linux-gnu/bin/arm-marvell-linux-gnu-gcc
LD= /usr/local/arm-marvell-linux-gnu/bin/arm-marvell-linux-gnu-ld
CFLAGS += -I/usr/local/arm-marvell-linux-gnu/include
LDFLAGS+=-L/usr/local/arm-marvell-linux-gnu/lib

all: $(EXEC)

$(EXEC): $(OBJS)
    $(CC) $(CFLAGS) $(OBJS) -o $@ $(LDFLAGS)
clean:
    rm -rf *.o $(PROG) *.core
```

Compiling open source projects

Most open source projects go through the following steps to compile the application:

1. configure
2. make
3. make install

As you know, “configure” is for determine the capability of a program. In most cases, to compile programs for other target machines requires that you modify the “configure” file manually to provide the correct values.

When running “configure” to configure the software package for cross compiling, you will need to specify the CC, LD, CFLAGS, host, target, and build, etc.

For example, for Marvell 5281 platform:

```
# env
CC=/usr/local/arm-marvell-linux-gnu/bin/arm-marvell-linux-gnu-gcc \
LD=/usr/local/arm-marvell-linux-gnu/bin/arm-marvell-linux-gnu-ld \
RANLIB=/usr/local/arm-marvell-linux-gnu/bin/arm-marvell-linux-gnu-ranlib \
CFLAGS="-I/usr/local/arm-marvell-linux-gnu/include" \
LDFLAGS="-L/usr/local/arm-marvell-linux-gnu/lib" \
./configure \
--host=armle-unknown-linux \
--target=armle-unknown-linux \
--build=i686-pc-linux \
--prefix=/usr/local
```

For Power PC8241 platform:

```
# env CC=/usr/local/powerpc-linux/bin/powerpc-linux-gcc \
LD=/usr/local/powerpc-linux/bin/powerpc-linux-ld \
RANLIB=/usr/local/powerpc-linux/bin/powerpc-linux-ranlib \
CFLAGS="-I/usr/local/powerpc-linux/include" \
LDFLAGS="-L/usr/local/powerpc-linux/lib" \
./configure \
--host=powerpc-unknown-linux \
--target=powerpc-unknown-linux \
--build=i686-pc-linux \
--prefix=/usr/local
```

For PowerPC 8544/8533 platform:

```
# env CC=/usr/local/powerpc-linux-gnuspe/bin/powerpc-linux-gnuspe-gcc \
\
LD=/usr/local/powerpc-linux-gnuspe/bin/powerpc-linux-gnuspe-ld \
RANLIB=/usr/local/powerpc-linux-gnuspe/bin/powerpc-linux-gnuspe-ranlib \
CFLAGS="-I/usr/local/powerpc-linux-gnuspe/include -mcpu=8548 -mhard-float -mfloat-gprs=double" \
LDFLAGS="-L/usr/local/powerpc-linux-gnuspe/lib" \
./configure \
--host=powerpc-unknown-linux \
--target=powerpc-unknown-linux \
--build=i686-pc-linux \
--prefix=/usr/local
```

For Marvell 6281 platform:

```
# env
CC=/usr/local/arm-none-linux-gnueabi/bin/arm-none-linux-gnueabi-gcc \
LD=/usr/local/arm-none-linux-gnueabi/bin/arm-none-linux-gnueabi-ld \
RANLIB=/usr/local/arm-none-linux-gnueabi/bin/arm-none-linux-gnueabi-ranlib \
CFLAGS="-I/usr/local/arm-none-linux-gnueabi/include" \
LDFLAGS="-L/usr/local/arm-none-linux-gnueabi/lib" \
./configure \
--host=armle-unknown-linux \
--target=armle-unknown-linux \
--build=i686-pc-linux \
--prefix=/usr/local
```

For Intel X86 platform:

```
# env CC=/usr/local/i686-linux-gnu/bin/i686-linux-gnu-gcc \
LD=/usr/local/i686-linux-gnu/bin/i686-linux-gnu-ld \
RANLIB=/usr/local/i686-linux-gnu/bin/i686-linux-gnu-ranlib \
CFLAGS="-I/usr/local/i686-linux-gnu/include" \
LDFLAGS="-L/usr/local/i686-linux-gnu/lib" \
./configure \
```

```

--host=i686-linux-gnu \
--target=i686-linux-gnu \
--build=i686-pc-linux \
--prefix=/usr/local

```

Compiling kernel module

If you want to compile kernel modules, you will need to obtain the Synology GPL to access the kernel source code. Please refer to <http://www.synology.com/enu/gpl/> for details.

In the kernel source, there are different configuration files for different platforms. The configuration file used in each model is listed below:

Model	Configuration file	Arch	Linux
CS407, DS107+, DS207+, RS407	88f5281-config	ARM	2.6.15
DS109, DS209, DS409, DS409slim, DS509, RS409	synoconfigs/88f6281	ARM	2.6.32
DS110j, DS210j, DS410j, DS211j, DS411j	synoconfigs/88f6281	ARM	2.6.32
DS111, DS211	synoconfigs/88f6281	ARM	2.6.32
CS407e, DS207, DS209j	synoconfigs/ppc824x	PowerPC	2.6.24
DS107e, DS107, DS108j, DS109j	powerpc-config	PowerPC	2.4
DS508, RS408, RS408-RP	synoconfigs/ppc854x	PowerPC	2.6.32
DS109+, DS209+, DS409+, RS409+, RS409RP+	synoconfigs/ppc8533	PowerPC	2.6.32
DS110+, DS210+, DS410	synoconfigs/ppc8533	PowerPC	2.6.32
DS710+, DS1010+, DS411+, RS810+, RS810RP+	synoconfigs/x86	x86	2.6.32

Please copy the proper configuration file to “.config” and run “make oldconfig”, and “make menuconfig” to select your kernel modules. Depending on the platform you want to compile, you have to set the proper ARCH and CROSS_COMPILE into environment variables.

```
For example, to compile 2.6 kernel modules for 5281 platform:

# cd linux-2.6.15
# cp 88f5182-config .config
# make ARCH=arm \
CROSS_COMPILE=/usr/local/arm-marvell-linux-gnu/bin/arm-marvell-linux
-gnu- oldconfig
# make ARCH=arm \
CROSS_COMPILE=/usr/local/arm-marvell-linux-gnu/bin/arm-marvell-linux
-gnu- menuconfig
# make ARCH=arm \
CROSS_COMPILE=/usr/local/arm-marvell-linux-gnu/bin/arm-marvell-linux
-gnu- modules
```

For 2.4 kernel:

```
# cd uclinux2422/linux-2.4.x
# cp powerpc-config .config
# cp Makefile.powerpc Makefile
# make oldconfig
# make menuconfig (and choose your modules)
# make dep
# make modules
```

Installation

After compiling the open source package or your own application, the “make install” will not install the application on the Synology DiskStation. Instead, it will install the application to your Linux PC. What you have to do is to copy the needed files to the Synology DiskStation. Details are described in the following section.

Put the application in `/usr/local`

Synology releases new DSM from time to time. It is important that you install your application in the correct directory so that it will not be deleted when DSM upgrades.

The `/usr/local` is reserved for 3rd-party application. It is recommended that you create a directory for your application and put related files in it. For example, you can create `/usr/local/myapp` and then create a directory `bin` in it for utilities, `sbin` for daemons and system utilities, `etc` for configuration files, and `lib` for your libraries. After done, you can copy the related files of your application into the respective directories. Please note that you may need to specify the correct prefix when running “configure”, so that the application can find the correct path information upon execution.

When performing DSM upgrade, the directory `/usr/local` will be backed up and restored automatically. However, some libraries files or built-in software packages may be modified during the upgrade procedure, so if your application depends on these files, it may not run after.

Run the application when the system boots

If you would like to run the application when the system boots up, you have to write a startup script and put it in `/usr/local/etc/rc.d/`. There are some rules for the startup script:

1. It must have suffix “.sh”. For example, “**myprog.sh**”.
2. The permission must be 755.
3. It must take the options “start” and “stop”. When the system boots up, it will call “**myprog.sh start**”; when system shuts down, it will call “**myprog.sh stop**”.

You can refer to the scripts in **`/usr/syno/etc/rc.d/`**. They are scripts for Synology default services.

Integrating into the Synology DiskStation Manager 2.3

The Synology DiskStation Manager 2.3 can integrate 3rd-party applications into its new AJAX management UI as a menu item. You can also customize its icon and put it on the management desktop.

For example, the following picture shows a 3rd-party application named "MyApp". When clicking on it, the Synology official website will be displayed in the right frame.



Startup

To integrate an application into the Synology DiskStation Manager 2.3, follow the steps below:

1. Create a directory in directory `/usr/syno/synoman/webman/3rdparty/`.
2. Create a text file named "**application.cfg**" under the directory. (Details will be given in next section.)
3. Under the same directory, you can create a sub-directory named by your application, like `/usr/syno/synoman/webman/3rdparty/myapp/`, for example. You can put all UI related components, such as images, CSS, and CGI in the directory.

Next we will show the details of UI configuration.

Application.cfg

“**application.cfg**” is a text file for configuring UI behavior. Please note the encoding of “**application.cfg**” should be in UTF-8 for the description being shown correctly on the web interface.

The content of “**application.cfg**” should be in **key=value** format. For example:

```
text = My app
description = This is my menu item pointed to Synology WebSite
icon_16 = images/icon16.png
icon_32 = images/icon32.png
type = embedded
protocol = http
address = www.synology.com
port = 80
path = /index.php
```

Below lists more information about the contents of **application.cfg**:

Key	Description
text (required)	<p>“text” describes the menu item names shown in the menu tree.</p> <p>If you would like to localize the menu items, you can add a language abbreviation suffix to the file.</p> <p>For example:</p> <p style="padding-left: 40px;"><code>text_cht</code> → Menu item name in Traditional Chinese</p> <p style="padding-left: 40px;"><code>text_fre</code> → Menu items name in French</p> <p>The abbreviation for other language (e.g. jpn, sve, spn...) can be found at /usr/syno/synoman/webman/texts/. If user uses language that is not available here, "text" will be used by default. It is not recommended to create language abbreviations not included in the list. The system may not recognize these abbreviations.</p>
description (required)	<p>“description” describes the details of the menu item. It will show upon mouse-over and in the complete function list. This string can also be localized by adding language abbreviation suffix to the file.</p> <p>For example:</p> <p style="padding-left: 40px;"><code>description_cht</code></p> <p style="padding-left: 40px;"><code>description_fre</code></p>
icon_16	<p>“icon_16” describes menu item icon in 16x16 size. The icon size should be 16x16 pixels and format in .png.</p> <p>The icon must be put under /usr/syno/synoman/webman/3rdparty/MyApp/.</p> <p>For example, if you want to use the image “icon.png” as the icon of your application, you should set “icon_16” value should as “images/icon.png” in the application.cfg file.</p> <p>If you create a directory named "images" and put all related images files of the</p>

	<p>application in it, the full path for your icon file "icon.png" should be:</p> <p>/usr/syno/synoman/webman/3rdparty/xxx/images/icon.png</p>
icon_32	<p>This is the key for large icon. It is a 32x32 .PNG file. This icon will be used in complete function list mode. Its value settings are the same as icon_16.</p>
type	<p>When you click on the menu item, the address you use to connect to the DSM management UI will be shown in the right frame of the management UI. However, you can customize the address as you wish.</p> <p>"type" describes the way how the URL is shown. Its value can be "embedded" or "popup". "popup" means when you click on the menu item, the URL will be opened in a pop-up window. "embedded" means when you click on the menu item, the URL will be opened in the right management UI frame.</p> <p>If the "type" is not specified, "popup" will be the default type.</p> <p>You can follow the description below to set up your customized URL.</p>
protocol	<p>The value of protocol can be "http" or "https", depending on the URL protocol you choose. If this value is not set, the current connection protocol will be used.</p> <p>protocol = http</p>
address	<p>This key describes the address of the URL you choose to link when you click on the menu item. If it is left empty, the address will be the address users use to connect to the DSM management UI.</p>
port	<p>This is the port number of the URL. If this is not specified, the port users use to connect to the management UI will be used.</p>
path	<p>This describes the path where you put the URL files. The value should always start with a "/".</p>
adminonly	<p>This key describes whether the menu items only show when users login by admin account. If you would like all users to be able to see the menu item, please set the key value as below:</p> <p>adminonly = false</p> <p>The default setting is that only the admin can see the menu item.</p>

Example

Example 1: If you want to add a menu item named "My Menu Item", and pops up a new window of <http://www.synology.com/index.php> when you click on it, follow the steps below:

1. Create the directory **/usr/syno/synoman/webman/3rdparty/my_menu_item**.
2. Put the icon file in **/usr/syno/synoman/webman/3rdparty/my_menu_item/images/**.

- Put the “**application.cfg**” in **/usr/syno/synoman/webman/3rdparty/**. The contents of “application.cfg” should be like this:

```
text = My Menu Item
description = This is my menu item pointed to Synology Website
icon_16 = images/icon16.png
icon_32 = images/icon32.png
type = popup
protocol = http
address = www.synology.com
port = 80
path = /index.php
```

Example 2: If you want to add a menu item named "Second Menu Item", and display Synology Web Station (port 80) of the Synology DiskStation when you click on it, follow the steps below:

- Create a directory **/usr/syno/synoman/webman/3rdparty/**.
- Put the file “**application.cfg**” in it. The “**application.cfg**” should be like this:

```
text = Second Menu Item
description = Description of second menu item
type = embedded
protocol = http
port = 80
```

Here the “address” is not set, which means the default URL (Synology official website) will be shown as embedded (in the right management UI frame). The icon is not specified, either. So the default icon will be used.

Adding Your Application as Desktop Item

The Desktop page is the new feature introduced in Synology DSM 2.3. It allows users to access applications with one click. It even allows users to add their customized 3rd-party application icons on it.



To add an application icon on the Desktop of the Synology DSM 2.3, follow the steps below:

1. Create a directory for your application under the directory **/usr/syno/synoman/webman/3rdparty/**. For example, you can create a directory named "SqueezeCenter" for the application. The path should be like this:
/usr/syno/synoman/webman/3rdparty/SqueezeCenter
2. To add the icon of the application on the desktop, you should put a text file named "**desktop.cfg**" and the image file of the icon under the directory. The icon will be thus customized.

Below describes the detailed contents of the **desktop.cfg** file.

Desktop.cfg

The “**desktop.cfg**” is a text file for configuring the desktop icon and application link. Please note the encoding of “**desktop.cfg**” should be in UTF-8 for the descriptions being shown correctly on the web interface.

The content of “**desktop.cfg**” should be in **key=value** format. For example:

```
adminonly=false
text=SqueezeCenter
description=SqueezeCenter Music Player
protocol=http
path=/
port=9001
icon=images/desktop.png
icon_alt=images/desktop_alt.png
```

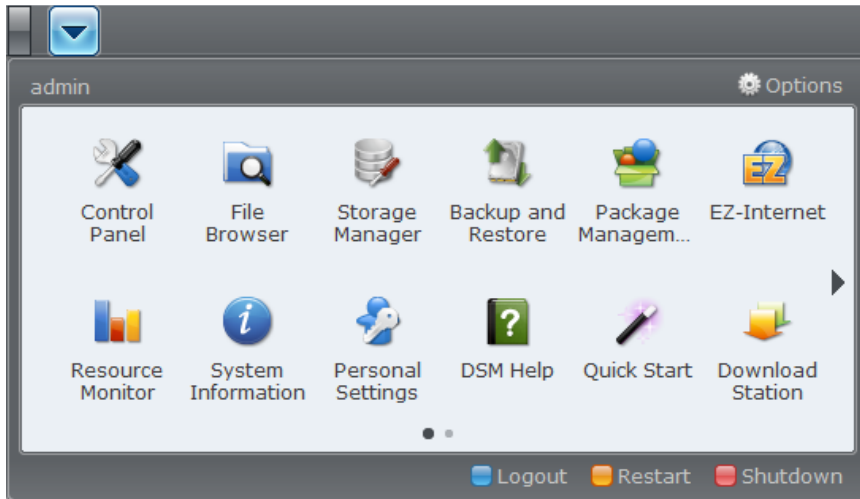
Here are the details for the content of application.cfg:

Key	Description
text (required)	<p>“text” describes the menu item names shown in the menu tree. It usually shows under the icon.</p> <p>If you would like to localize the menu items, you can add a language abbreviation suffix to the file.</p> <p>For example:</p> <p style="padding-left: 40px;"><code>text_cht</code> → Menu item name in Traditional Chinese</p> <p style="padding-left: 40px;"><code>text_fre</code> → Menu items name in French</p> <p>The abbreviation for other language (e.g. jpn, sve, spn...) can be found at /usr/syno/synoman/webman/texts/. If user uses language that is not available here, "text" will be used by default. It is not recommended to create language abbreviations not included in the list. The system may not recognize these abbreviations.</p>
description (required)	<p>“description” describes the details of the application. It will show upon mouse-over and in the complete function list. This string can also be localized by adding language abbreviation suffix to the file.</p> <p>For example:</p> <p style="padding-left: 40px;"><code>description_cht</code></p> <p style="padding-left: 40px;"><code>description_fre</code></p>
icon (required)	<p>“icon” describes the application icon on the desktop. It should be an image file of size 117x87 pixels and in .PNG format.</p> <p>The icon must be put under /usr/syno/synoman/webman/3rdparty/MyApp/. You can customize MyApp as you wish.</p> <p>For example, if you want to use the image “icon.png” as the icon of your application, you should set “icon_16” value as "images/icon.png" in the application.cfg file.</p>

	<p>If you create a directory named "images" and put all related images files of the application in it, the full path for your icon file "icon.png" should be:</p> <p><code>/usr/syno/synoman/webman/3rdparty/xxx/images/icon.png</code></p>
icon_alt	<p>This key describes the alternative icon when mouse-over on the desktop icon. Its value settings are the same as "icon".</p>
protocol	<p>When you click on the menu item, the address you use to connect to the DSM management UI will be shown in the right frame of the management UI. However, you can customize the address as you wish.</p> <p>The value of protocol can be "http" or "https", depending on the URL protocol you choose. If this value is not set, the current connection protocol will be used.</p> <p><code>protocol=http</code></p>
address	<p>This key describes the address of the URL you choose to link when you click on the application icon. If it is left empty, the address will be the address users use to connect to the DSM management UI.</p>
port	<p>This is the port number of the URL. If this is not specified, the port users use to connect to the management UI will be used.</p>
path	<p>This describes the path where you put the URL files. The value should always start with a "/".</p>
adminonly	<p>This key describes whether the desktop icon only show when users login by admin account. If you would like that all users to be able to see the menu item, please set the key value as below:</p> <p><code>adminonly = false</code></p> <p>The default setting is that only the admin can see the menu item.</p>

Integrating into the Synology DiskStation Manager 3.0

In Synology DSM 3.0, adding 3rd-party applications on your DiskStation is even easier than before. When an application is integrated, its icon will be automatically added in the Main Menu of DSM 3.0. Users can also use customized icons for these applications. To add the icon on the desktop, you only have to drag it from the Main Menu to the Desktop on DSM.



Startup

To integrate an application into the Synology DiskStation Manager 3.0, follow the steps below:

1. Create a directory in the `/usr/syno/synoman/webman/3rdparty/`.
2. Create a text file named "**config**" under the directory.
3. Under the same directory, you can create a sub-directory named by your application, like `/usr/syno/synoman/webman/3rdparty/myapp/`, for example. You can put all UI related components, such as images, CSS, and CGI in the directory.

Next we will show the details of UI configuration.

config

The “**config**” is a text file to configure the UI behavior. The content of “**config**” should be in **JSON** format. For example:

```
{
  ".url": {
    "com.company.App1": {
      "type": "url",
      "allUsers": true,
      "title": "Test App1",
      "desc": "Description",
      "icon": "images/app_{0}.png",
      "url": "http://www.yahoo.com"
    },
    "com.company.App2": {
      "type": "legacy",
      "allUsers": true,
      "title": "Test App2",
      "desc": "Description 2",
      "icon": "images/app2_{0}.png",
      "url": "http://www.synology.com"
    }
  }
}
```

Here are the details for the content of application.cfg:

Property	Description
com.company.App1 com.company.App2	In “.url”, each object should have a unique property name.
title (Required)	“title” describes the application name that will be shown in the main menu.
desc	“desc” describes more details about this application upon mouse-over.

icon (Required)	<p>"icon" describes the icon for the application. It is a template string. The "{0}" can be replaced by "16", "24", "32", "48", depending on different pixels of the image file for the icon.</p> <p>The icon must be put under /usr/syno/synoman/webman/3rdparty/xxx/ where xxx is the directory name of your application.</p> <p>For example, if you create a directory named "images" and put the icon image file "icon.png" in it, the full path for the icon will be:</p> <p>/usr/syno/synoman/webman/3rdparty/xxx/images/icon_16.png /usr/syno/synoman/webman/3rdparty/xxx/images/icon_24.png /usr/syno/synoman/webman/3rdparty/xxx/images/icon_32.png /usr/syno/synoman/webman/3rdparty/xxx/images/icon_48.png</p> <p>And the icon value should be set to "images/icon_{0}.png".</p>
type (Required)	<p>When you click on the menu item, the address you use to connect to the DSM management UI will be shown in the right frame of the management UI. However, you can customize the address as you wish.</p> <p>The "type" value can be "url" or "legacy". "url" means that when clicking on the application icon, the URL will be opened in a pop-up window.. while "legacy" means when you click on the menu item, the URL will be opened in an iframe window application.</p> <p>You can follow the description below to set up your customized URL.</p>
url (Required)	<p>This is an example of value setting for your URL of the application:</p> <pre> "url": "http://www.synology.com/" "url": "3rdparty/xxx/index.html" </pre>
allUsers	<p>This key describes if the application is only shown when users login by admin account. If you would like that all users can see the menu item, please set the key value as below:</p> <pre> "allUsers": true </pre> <p>The default setting is that only the admin can find the application.</p>

Integrating with Synology web authentication

After integrating your application into the Synology DSM, you may want to perform an authentication check to ensure only logged in users can access the page.

To check whether a user has logged in, you can run the command below in the CGI:

```
/usr/syno/synoman/webman/modules/authenticate.cgi
```

The “**authenticate.cgi**” will output the user name if the user has logged in. There will not be any output if the user has not been authenticated.

Below is an example:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <strings.h>

/**
 * Check whether user is logged in.
 *
 * If user has logged in, put the username into "user".
 *
 * @param user   The buffer for get username
 * @param bufsize The buffer size of user
 *
 * @return 0: User not logged in or error
 *         1: User logged in. The user name is written to given "user"
 */
int IsUserLogin(char *user, int bufsize)
{
    FILE *fp = NULL;
    char buf[1024];
    int login = 0;

    bzero(user, bufsize);

    fp = popen("/usr/syno/synoman/webman/modules/authenticate.cgi",
"r");
    if (!fp) {
        return 0;
    }
    bzero(buf, sizeof(buf));
    fread(buf, 1024, 1, fp);

    if (strlen(buf) > 0) {
        snprintf(user, bufsize, "%s", buf);
        login = 1;
    }
    pclose(fp);

    return login;
}
```

```
}  
  
int main(int argc, char **argv)  
{  
    char user[256];  
  
    printf("Content-type: text/html\r\n\r\n");  
    if (IsUserLogin(user, sizeof(user)) == 1) {  
        printf("User is authenticated. Name: %s\n", user);  
    } else {  
        printf("User is not authenticated.\n");  
    }  
    return 0;  
}
```

This CGI runs the command below to check whether a user is logged in. After checking up, it will print out the user name if the user has logged in.

```
/usr/syno/synoman/webman/modules/authenticate.cgi
```

Document Revision History

This table describes the changes to the *Synology DiskStation Manager 3rd-Party Apps Developer Guide*.

Date	Note
2008-06-16	1. Document originally released
2009-02-09	1. Add Freescale 8533 tool chain information
2009-03-	1. Add Marvell 6281 tool chain information 2. Add Desktop Icon chapter
2010-05-20	1. Add related information for 10 models 2. Change all DSM 2.0 & 2.1 to DSM 2.3 3. New Screen shots for desktop icon & DSM application 4. Change Configuration files and Tool chains 5. DS1010+ uses Intel Atom D510
2010-05-28	1. Fix typo 2. Revise Tool Chain description 3. Revise 88f5182-config to 88f5281-config in Kernel Module 4. Revise path description of application.cfg / desktop.cfg
2010-11-29	1. Rename the document to be "Synology DiskStation Manager 3rd-Party Apps Developer Guide" 2. Add DSM 3.0 Integration chapter