How To Make a MPB Package

From MOD Wiki

This page describes how to make a plugin package for mod-plugin-builder (or MPB for short)

Plugin packages reside in plugins/package/ directory.

Contents

Plugin package example

Here's an example of a plugin package file:

```
PLUGINPKG_VERSION = 1.0.0
PLUGINPKG_SITE = http://download.sourceforge.net/myplugin/
PLUGINPKG_SOURCE = myplugin-$(PLUGINPKG_VERSION).tar.gz
PLUGINPKG_BUNDLES = myplugin.lv2

$(eval $(cmake-package))
```

We're using PLUGINPKG as a generic name, **you must use your package name in uppercase** here.
Let's divide this into small pieces...

Version and download location

First we define the version, plus the download location and filename to download the source code from.

```
PLUGINPKG_VERSION = 1.0.0
PLUGINPKG_SITE = http://download.sourceforge.net/myplugin/
PLUGINPKG_SOURCE = myplugin-$(PLUGINPKG_VERSION).tar.gz
```

If you rather use a git repository, use something like this:

```
PLUGINPKG_VERSION = 25451be928b69c288f6978fb3b3fcf202dbd1ee1
PLUGINPKG_SITE = git://github.com/myself/myplugin
PLUGINPKG_SITE_METHOD = git
```

LV2 bundles

Moving on, we define which bundles to use.
Your build system can install more bundles than what's defined here, but only the defined ones will be picked up to be published locally and on the cloud.
Use a space to separate the bundle names. Newline escaping is not supported, everything must be in the same line.
(Note: they must be installed to $DESTDIR/usr/lib/lv2/)

```
PLUGINPKG_BUNDLES = mybundle1.lv2 mybundle2.lv2
```

Build rules

Finally, we defined the steps to build the package.
If you're using autotools or cmake, buildroot has this covered for you already.
Using other build systems means you have to specify how to configure, build and install the code.

autotools

For autotools, use:

```
$(eval $(autotools-package))
```

cmake

For cmake, use:

```
$(eval $(cmake-package))
```

waf

Here's an example for waf:

```
PLUGINPKG_TARGET_WAF = $(TARGET_MAKE_ENV) $(TARGET_CONFIGURE_OPTS) $(HOST_DIR)/usr/bin/python ./waf

define PLUGINPKG_CONFIGURE_CMDS
        (cd $(@D); $(PLUGINPKG_TARGET_WAF) configure --prefix=/usr)
endef

define PLUGINPKG_BUILD_CMDS
        (cd $(@D); $(PLUGINPKG_TARGET_WAF) build -j $(PARALLEL_JOBS))
endef

define PLUGINPKG_INSTALL_TARGET_CMDS
        (cd $(@D); $(PLUGINPKG_TARGET_WAF) install --destdir=$(TARGET_DIR))
endef
```

```
$(eval $(generic-package))
```

raw makefile

And here's an example for raw makefiles:

```
PLUGINPKG_TARGET_MAKE = $(TARGET_MAKE_ENV) $(TARGET_CONFIGURE_OPTS) $(MAKE) -C $(@D)

define PLUGINPKG_BUILD_CMDS
        $(PLUGINPKG_TARGET_MAKE)
endef

define PLUGINPKG_INSTALL_TARGET_CMDS
        $(PLUGINPKG_TARGET_MAKE) install DESTDIR=$(TARGET_DIR)
endef

$(eval $(generic-package))
```

Tips and tricks

If using autotools or cmake and you need to specify options during configure, use something like this:

```
PLUGINPKG_CONF_OPTS=-DBUILD_GUI=OFF
```

If using autotools without an existing 'configure' script (ie, autogen.sh needs to be run first), use this:

```
PLUGINPKG_AUTORECONF = YES
```

If using autotools, cmake or raw makefiles and multiple jobs breaks your build, use this:

```
PLUGINPKG_MAKE = $(MAKE1)
```

Final notes

Plugin packages are buildroot files. Because of that it must comply with Buildroot rules.
A few important notes:

- The package name is defined by the folder name and cannot contain '.'
- There must be a <packagename>.mk file inside the package folder
- The package name and '.mk' file name must be the same
- Inside the '.mk' file all defined variables must start with the package name in uppercase replacing '-' with '_'

Browse through other examples so you get an idea of variations of project files.

"https://wiki.moddevices.com/index.php?title=How_To_Make_a_MPB_Package&oldid=11347"
