MarsOS/Ship - Blueprint File and Package Specification

July 30, 2017

Abstract

This document describes the specification for the configuration files used for 'ship', the package manager of MarsOS and how the packages are build

1 Introduction

Blueprint files are, as their name suggests, a manual for the package manager. They describe how a package should be build and also define some meta-data which cannot be determined otherwise.
An example of this meta-data could be the packager name or the license of the package. Other meta-data, like the size, is generated automatically by the package manager during the build of a package.
A Blueprint file is encoded in JSON.

2 Blueprint File

All blueprint files must have the name 'blueprint.json'. This is required, so that the package manager can locate them automatically if it is in the source directionary. If you are not in the source, you can pass the path to the blueprint to the package manager to build the package (and download the sources to a temp folder).

3 Build Environment

The package manager must expose curtain variables during the build, to allow a blueprint file to be written in a portable way. You should never use an absolute path in a blueprint, because it will may break the build under curtain conditions. To avoid this, you should use one of the environment variables below (Shell-Syntax):

Variable	Description
$srcdir	Path to your source files (with trailing slash)
$builddir	install your files here, treat it like root (/)
$arch	The architecture on which the package is build, for example x86_64

4 Blueprint Object

As stated above, a blueprint is encoded in JSON. This section will document all available and required fileds in a blueprint file.

4.1 Required Entries

4.1.1 name

The name of the package. Should only contain (lowercase) letters, numbers and hyphens(-).

4.1.2 url

Homepage of the package, can also be an URL of the source repository.

4.1.3 getVersion

A shell command that will output the version of the current sources.

4.1.4 source

Where to download the sources. This is an actual object and needs to contain at least the 'latest' child, mapped to an URL to download the latest version but could also hold multiple versions like '1.0.2'

4.1.5 config

An array of commands to execute before the build process.
(e.g. ./configure))

4.2 build

An array of commands used to build the code. (e.g. make)

4.2.1 package

An array of commands used to package your code, normally this will be something like:
'make install DESTDIR=$pkgdir'

4.2.2 maintainer

An object with to child properties, name and email. It should contain information about the person who created the blueprint and should be contacted if there is an error.

4.2.3 license

The license of the package, can be anything from the following list:

- MIT
- (L)GPLv3
- (L)GPLv2
- Apache
- BSD (3-Clause)
- CC0 (public domain)
- other

4.3 Optional Entries

4.3.1 bugtracker

Where to report bugs, normally an URL or for small projects an email.

4.3.2 invalidArches

An array of architectures on which the package can not be build. You should really avoid this, but sometimes it is not possible to do so.

5 Example File

The following file is taken from the package manager itself, so yes, it can package itself.

```
{
        "name": "ship",
        "license": "MIT",
        "url": "https://github.com/MarsOS/ship",
        "bugtracker": "https://github.com/MarsOS/ship/issues",
        "getVersion": "make version",
        "source": {
        "latest": "https://github.com/MarsOS/ship/archive/master.tar.gz"
        },
        "config": [

        ],
        "build": [
                "make"
        ],
        "package": [
                "make install DESTDIR=$pkgdir"
        ],
        "maintainer": {
                "name": "Marius Messerschmidt",
                "email": "marius.messerschmidt@googlemail.com"
        }
}
```