

---

# **dmgbuild Documentation**

*Release 1.3.0*

**Alastair Houghton**

**Jul 27, 2017**



---

# Contents

---

<b>1</b>	<b>What is this?</b>	<b>3</b>
1.1	Usage . . . . .	3
1.2	Settings . . . . .	3
1.3	Example Settings File . . . . .	9
<b>2</b>	<b>Indices and tables</b>	<b>15</b>



This document refers to version 1.3.0



# CHAPTER 1

---

## What is this?

---

`dmgbuild` is a command line tool to create macOS disk images (aka `.dmg` files). While it is possible to create disk images easily enough from the command line using the `hdiutil` program that ships with macOS, there is no easy way to configure the appearance of the resulting disk image when the user opens it. Some people have used AppleScript to automate Finder to adjust the appearance, but since Finder saves its `.DS_Store` files asynchronously, it is hard to guarantee that the changes will actually be saved when you want them to be. It also means that you need a GUI session, with Finder running, in order to build your disk image.

`dmgbuild` does not rely on Finder; nor does it rely on deprecated APIs (like the Alias Manager functions). Instead, it uses the `ds_store` and `mac_alias` Python modules, which know how to construct the relevant data in Python code.

Contents:

## Usage

Typical usage looks like this:

```
dmgbuild -s settings.py "Volume Name" output.dmg
```

The (optional) `settings.py` file specifies the attributes that control the appearance of the disk image. The other arguments give the volume name and the name of the output file respectively.

`dmgbuild` also accepts arguments of the form `-D key=value`; these can be used from within the settings file, for instance so that you can write a single settings file for multiple disk images, or so that you can easily alter settings at build time.

## Settings

`dmgbuild` accepts, as one of its arguments, a settings file. This is in fact a Python script, which means anything you can do in Python code, you can do in your settings file. This makes it easy for you to customise the behaviour of `dmgbuild`.

Each of the available settings is documented below; all of them are optional; the `dmgbuild` program has defaults for those that matter. The default values *are* visible from within your settings file, if you want to examine or alter them rather than replacing them completely.

If any `-D key=value` settings have been made on the command line, they are visible in a dictionary named `defines` within the settings file.

## Disk Image Settings

### **filename**

If defined, overrides the output filename specified on the command line. The command line value is the default value.

### **volume\_name**

If defined, overrides the volume name specified on the command line, which is the default value.

### **format**

Specifies the format code for the final output disk image. Must be one of the types supported by `hdiutil` on the build system; currently the list includes

Code	Meaning
UDRO	Read-only
UDCO	Compressed (ADC)
UDZO	Compressed (gzip)
UDBZ	Compressed (bzip2)
UFBI	Entire device
IPOD	iPod image
UDxx	UDIF stub
UDSB	Sparse bundle
UDSP	Sparse
UDRW	Read/write
UDTO	DVD/CD master
DC42	Disk Copy 4.2
RdWr	NDIF read/write
Rdxx	NDIF read-only
ROCo	NDIF Compressed
Rken	NDIF Compressed (KenCode)

For disk images you intend to distribute over the Internet, you should probably stick to ‘UDZO’ and ‘UDBZ’.

### **size**

If defined, specifies the size of the filesystem within the image. If this is not defined, `dmgbuild` will attempt to determine a reasonable size for the image.

If you set this, you should set it large enough to hold the files you intend to copy into the image. The syntax is the same as for the `-size` argument to `hdiutil`, i.e. you can use the suffixes ‘b’, ‘k’, ‘m’, ‘g’, ‘t’, ‘p’ and ‘e’ for bytes, kilobytes, megabytes, gigabytes, terabytes, exabytes and petabytes respectively.

## Content Settings

### **files**

A list of files (or folders) to copy into the image. Each of these is copied to the root of the image; folders are copied recursively. e.g.:

```
files = [ '/Applications/TextEdit.app' ]
```

**symlinks**

A dictionary specifying symbolic links to create in the image. For example:

```
symlinks = { 'Applications': '/Applications' }
```

**icon**

Specifies the path of an icon file to copy to the volume. You can either specify this, or as an alternative you can use the *badge\_icon* setting.

**badge\_icon**

As an alternative to the above, if you set *badge\_icon* to the path of an icon file or image, it will be used to badge the system's standard external disk icon. This is a convenient way to construct a suitable icon from your application's icon, e.g.:

```
badge_icon = '/Applications/TextEdit.app/Contents/Resources/Edit.icns'
```

**icon\_locations**

A dictionary specifying the co-ordinates of items in the root directory of the disk image, where the keys are filenames and the values are (x, y) tuples. e.g.:

```
icon_locations = {
    'TextEdit.app': (100, 100),
    'Applications': (300, 100)
}
```

## Window Settings

**background**

A string containing any of the following:

Example	Meaning
#3344ff	Web-style RGB color
#34f	Web-style RGB color, short form (#34f = #3344ff)
rgb(1,0,0)	RGB color, each value is between 0 and 1
hsl(120,1,.5)	HSL (Hue Saturation Lightness) color
hwb(300,0,0)	HWB (Hue Whiteness Blackness) color
cmyk(0,1,0,0)	CMYK (Cyan Magenta Yellow Black) color
goldenrod	X11/SVG named color
builtin-arrow	A simple blue arrow image (retina enabled)
/foo/bar/baz.png	The path to an image file

The hue component in `hsl()` and `hwb()` may include a unit; it defaults to degrees ('deg'), but also supports radians ('rad') and gradians ('grad' or 'gon').

Other color components may be expressed either in the range 0 to 1, or as percentages (e.g. 60% is equivalent to 0.6).

For no background, specify `None` instead of a string value.

**show\_status\_bar****show\_tab\_view****show\_toolbar****show\_pathbar**

**show\_sidebar**

Each of the above controls the display of one of the standard window elements. All of them default to `False`.

**sidebar\_width**

The width of the Finder sidebar.

**window\_rect**

The position of the window in  $(x, y, w, h)$  format, with  $y$  co-ordinates running from bottom to top. The Finder makes sure that the window will be on the user's display, so if you want your window at the top left of the display you could use  $(0, 100000)$  as the  $x, y$  co-ordinates. Unfortunately it doesn't appear to be possible to position the window relative to the top left or relative to the centre of the user's screen.

**default\_view**

The default view for the window; should be a string containing one of:

View name
icon-view
list-view
column-view
coverflow

**show\_icon\_preview**

Whether or not to show icon previews for the contents of the disk image (defaults to `False`)

**include\_icon\_view\_settings**

**include\_list\_view\_settings**

Set these to `True` to force inclusion of the icon/list view settings respectively. By default, `dmgbuild` will only include settings for the default view type.

## Icon View Settings

**arrange\_by**

If set, indicates that the Finder should arrange the icons in the icon view according to the specified field. Allowable settings are:

Field name
name
date-modified
date-created
date-added
date-last-opened
size
kind
label

Any other value disables automatic icon arrangement (which is the default, since the main use-case for `dmgbuild` is building application distribution images, where icon positioning is an important part of the design).

**grid\_offset**

Specifies the grid offset for automatic arrangement.

**grid\_spacing**

Specifies the grid spacing for automatic arrangement.

**Warning:** As of Mac OS X Yosemite (v10.10), Finder checks to make sure that `grid_spacing` is less than 100. If it is over that value, it will reject the saved settings.

**scroll\_position**

An (x, y) tuple specifying the scroll position; this is only relevant if you position icons outside of the window area.

**label\_pos**

Specifies the position of the icons' labels. Choose 'bottom' or 'right' (defaults to 'bottom').

**text\_size**

Specifies the point size of the label text. Default is 16pt.

**icon\_size**

Specifies the size of icon to use. Default is 128pt.

**icon\_locations**

If `arrange_by` is not set, a dictionary mapping the names of items in the root of the volume to an (x, y) tuple specifying their location in points.

## List View Settings

In list view, the following columns are available:

Field name
name
date-modified
date-created
date-added
date-last-opened
size
kind
label
version
comments

**list\_icon\_size**

Sets the size of the icon in list view. Default is 16pt.

**list\_text\_size**

Sets the size of the text in list view. Default is 12pt.

**list\_scroll\_position**

Specifies the scroll position, assuming there are enough items to make the view scroll.

**list\_sort\_by**

Specifies which column the Finder should sort the display by. Defaults to 'name'.

**list\_use\_relative\_dates**

If `True`, formats dates using words like "Today" or "Yesterday" where possible; otherwise they will be displayed as a full date. Defaults to `True`.

**list\_calculate\_all\_sizes**

If `True`, forces the Finder to compute all of the item sizes; normally this is set to `False` because it can be expensive calculating the sizes of deeply nested folders. Defaults to `False`.

**list\_columns**

A list or tuple of strings containing the names of columns, in the order you want them to appear.

**list\_column\_widths**

A dictionary specifying the width, in points, for each of the columns. There are default widths for every column, so you may not need to set this variable in practice.

**list\_column\_sort\_directions**

A dictionary specifying the sort direction (either ‘ascending’, or ‘descending’) for each column. Again, there are individual defaults for each column, so you may not need to touch this unless you wish to override the default behaviour.

## License Settings

dmgbuild can attach license text to your disk image; this will be displayed automatically when the user tries to open your disk image.

Note that license text is either RTF, or it must be encoded in the legacy Mac encoding matching its language; dmgbuild will *try* to do this, but the built-in set of codecs in Python doesn’t cover all the Mac encodings, so in some cases you will need to encode the data and use a byte string.

**license**

If defined, a dictionary specifying the details of the license to display. It has the following keys:

Key	Op-tional	Value
default-language	No	The name of the default language to display if there is no license matching the system language.
licenses	No	A dictionary mapping language names to license text (either RTF data or plain text) or paths to files containing the license text.
buttons	Yes	A dictionary mapping language names to a sequence of user interface strings.

Recognized languages are:

af\_ZA, ar, be\_BY, bg\_BG, bn, bo, br, ca\_ES, cs\_CZ, cy, da\_DK, de\_AT, de\_CH, de\_DE, dz\_BT, el\_CY, el\_GR, en\_AU, en\_CA, en\_GB, en\_IE, en\_SG, en\_US, eo, es\_419, es\_ES, et\_EE, fa\_IR, fi\_FI, fo\_FO, fr\_001, fr\_BE, fr\_CA, fr\_CH, fr\_FR, ga-Latg\_IE, ga\_IE, gd, grc, gu\_IN, gv, he\_IL, hi\_IN, hr\_HR, hu\_HU, hy\_AM, is\_IS, it\_CH, it\_IT, iu\_CA, ja\_JP, ka\_GE, kl, ko\_KR, lt\_LT, lv\_LV, mk\_MK, mr\_IN, mt\_MT, nb\_NO, ne\_NP, nl\_BE, nl\_NL, nn\_NO, pa, pl\_PL, pt\_BR, pt\_PT, ro\_RO, ru\_RU, se, sk\_SK, sl\_SI, sr\_RS, sv\_SE, th\_TH, to\_TO, tr\_TR, uk\_UA, ur\_IN, ur\_PK, uz\_UZ, vi\_VN, zh\_CN, zh\_TW

The user interface strings are as follows:

In- dex	Comment	Typical English text
0	Language name	English
1	Agree button label	Agree
2	Disagree button label	Disagree
3	Print button label	Print
4	Save button label	Save
5	Instruction text	If you agree with the terms of this license, press “Agree” to install the software. If you do not agree, press “Disagree”.

There are built-in user interface strings for the following languages:

English (en\_US), German (de\_DE), Spanish (es\_ES), French (fr\_FR), Italian (it\_IT), Japanese (ja\_JP), Dutch (nl\_NL), Swedish (sv\_SE), Brazilian Portuguese (pt\_BR), Simplified Chinese (zh\_CN), Traditional Chinese (zh\_TW), Danish (da\_DK), Finnish (fi\_FI), Korean (ko\_KR), Norwegian (nb\_NO)

For other languages, if you don’t specify a suitable set, dmgbuild will use the English defaults instead.

dmgbuild will auto-detect RTF data by looking for the string `{\rtf1` at the start of the data. If it does not find this string, it will assume that you have supplied plain text.

## Example Settings File

Below is a copy of the example settings file that you can find in the source distribution in the folder “example”.

```
# -*- coding: utf-8 -*-
from __future__ import unicode_literals

import biplist
import os.path

#
# Example settings file for dmgbuild
#

# Use like this: dmgbuild -s settings.py "Test Volume" test.dmg

# You can actually use this file for your own application (not just TextEdit)
# by doing e.g.
#
#   dmgbuild -s settings.py -D app=/path/to/My.app "My Application" MyApp.dmg

# .. Useful stuff .....

application = defines.get('app', '/Applications/TextEdit.app')
appname = os.path.basename(application)

def icon_from_app(app_path):
    plist_path = os.path.join(app_path, 'Contents', 'Info.plist')
    plist = biplist.readPlist(plist_path)
```

```

    icon_name = plist['CFBundleIconFile']
    icon_root, icon_ext = os.path.splitext(icon_name)
    if not icon_ext:
        icon_ext = '.icns'
    icon_name = icon_root + icon_ext
    return os.path.join(app_path, 'Contents', 'Resources', icon_name)

# .. Basics .....

# Uncomment to override the output filename
# filename = 'test.dmg'

# Uncomment to override the output volume name
# volume_name = 'Test'

# Volume format (see hdiutil create -help)
format = defines.get('format', 'UDBZ')

# Volume size
size = defines.get('size', None)

# Files to include
files = [ application ]

# Symlinks to create
symlinks = { 'Applications': '/Applications' }

# Volume icon
#
# You can either define icon, in which case that icon file will be copied to the
# image, *or* you can define badge_icon, in which case the icon file you specify
# will be used to badge the system's Removable Disk icon
#
#icon = '/path/to/icon.icns'
badge_icon = icon_from_app(application)

# Where to put the icons
icon_locations = {
    appname: (140, 120),
    'Applications': (500, 120)
}

# .. Window configuration .....

# Background
#
# This is a STRING containing any of the following:
#
# #3344ff          - web-style RGB color
# #34f             - web-style RGB color, short form (#34f == #3344ff)
# rgb(1,0,0)      - RGB color, each value is between 0 and 1
# hsl(120,1,.5)   - HSL (hue saturation lightness) color
# hwb(300,0,0)    - HWB (hue whiteness blackness) color
# cmyk(0,1,0,0)  - CMYK color
# goldenrod       - X11/SVG named color
# builtin-arrow   - A simple built-in background with a blue arrow
# /foo/bar/baz.png - The path to an image file
#

```

```

# The hue component in hsl() and hwb() may include a unit; it defaults to
# degrees ('deg'), but also supports radians ('rad') and gradians ('grad'
# or 'gon').
#
# Other color components may be expressed either in the range 0 to 1, or
# as percentages (e.g. 60% is equivalent to 0.6).
background = 'builtin-arrow'

show_status_bar = False
show_tab_view = False
show_toolbar = False
show_pathbar = False
show_sidebar = False
sidebar_width = 180

# Window position in ((x, y), (w, h)) format
window_rect = ((100, 100), (640, 280))

# Select the default view; must be one of
#
#   'icon-view'
#   'list-view'
#   'column-view'
#   'coverflow'
#
default_view = 'icon-view'

# General view configuration
show_icon_preview = False

# Set these to True to force inclusion of icon/list view settings (otherwise
# we only include settings for the default view)
include_icon_view_settings = 'auto'
include_list_view_settings = 'auto'

# .. Icon view configuration .....

arrange_by = None
grid_offset = (0, 0)
grid_spacing = 100
scroll_position = (0, 0)
label_pos = 'bottom' # or 'right'
text_size = 16
icon_size = 128

# .. List view configuration .....

# Column names are as follows:
#
#   name
#   date-modified
#   date-created
#   date-added
#   date-last-opened
#   size
#   kind
#   label
#   version

```

```

#   comments
#
list_icon_size = 16
list_text_size = 12
list_scroll_position = (0, 0)
list_sort_by = 'name'
list_use_relative_dates = True
list_calculate_all_sizes = False,
list_columns = ('name', 'date-modified', 'size', 'kind', 'date-added')
list_column_widths = {
    'name': 300,
    'date-modified': 181,
    'date-created': 181,
    'date-added': 181,
    'date-last-opened': 181,
    'size': 97,
    'kind': 115,
    'label': 100,
    'version': 75,
    'comments': 300,
}
list_column_sort_directions = {
    'name': 'ascending',
    'date-modified': 'descending',
    'date-created': 'descending',
    'date-added': 'descending',
    'date-last-opened': 'descending',
    'size': 'descending',
    'kind': 'ascending',
    'label': 'ascending',
    'version': 'ascending',
    'comments': 'ascending',
}

# .. License configuration .....

# Text in the license configuration is stored in the resources, which means
# it gets stored in a legacy Mac encoding according to the language.  dmgbuid
# will *try* to convert Unicode strings to the appropriate encoding, *but*
# you should be aware that Python doesn't support all of the necessary encodings;
# in many cases you will need to encode the text yourself and use byte strings
# instead here.

# Recognized language names are:
#
# af_ZA, ar, be_BY, bg_BG, bn, bo, br, ca_ES, cs_CZ, cy, da_DK, de_AT, de_CH,
# de_DE, dz_BT, el_CY, el_GR, en_AU, en_CA, en_GB, en_IE, en_SG, en_US, eo,
# es_419, es_ES, et_EE, fa_IR, fi_FI, fo_FO, fr_001, fr_BE, fr_CA, fr_CH,
# fr_FR, ga-Latg_IE, ga_IE, gd, grc, gu_IN, gv, he_IL, hi_IN, hr_HR, hu_HU,
# hy_AM, is_IS, it_CH, it_IT, iu_CA, ja_JP, ka_GE, kl, ko_KR, lt_LT, lv_LV,
# mk_MK, mr_IN, mt_MT, nb_NO, ne_NP, nl_BE, nl_NL, nn_NO, pa, pl_PL, pt_BR,
# pt_PT, ro_RO, ru_RU, se, sk_SK, sl_SI, sr_RS, sv_SE, th_TH, to_TO, tr_TR,
# uk_UA, ur_IN, ur_PK, uz_UZ, vi_VN, zh_CN, zh_TW

# license = {
#     'default-language': 'en_US',
#     'licenses': {
#         # For each language, the text of the license.  This can be plain text,

```

```

#      # RTF (in which case it must start "{\rtf1"), or a path to a file
#      # containing the license text.  If you're using RTF,
#      # watch out for Python escaping (or read it from a file).
#      'English': b'''\{\rtf1\ansi\ansicpg1252\cocoartf1504\cocoasubrtf820
# {\fonttbl{\f0\fnil\fcharset0 Helvetica-Bold;\f1\fnil\fcharset0 Helvetica;}
# {\colortbl;\red255\green255\blue255;\red0\green0\blue0;}
# {\*\expandedcolortbl;;\cssrgb\c0\c0\c0;}
# \paperw11905\paperh16837\margl1133\margr1133\margb1133\margt1133
# \deftab720
# \pard\pardefstab720\sa160\partightenfactor0

# \f0\b\fs60 \cf2 \expnd0\expndtw0\kerning0
# \up0 \nosupersub \ulnone \outl0\strokewidth0 \strokec2 Test License\
# \pard\pardefstab720\sa160\partightenfactor0

# \fs36 \cf2 \strokec2 What is this?\
# \pard\pardefstab720\sa160\partightenfactor0

# \f1\b0\fs22 \cf2 \strokec2 This is the English license. It says what you are_
↪allowed to do with this software.\
# \
# }'',
# },
# 'buttons': {
#     # For each language, text for the buttons on the licensing window.
#     #
#     # Default buttons and text are built-in for the following languages:
#     #
#     #   English (en_US), German (de_DE), Spanish (es_ES), French (fr_FR),
#     #   Italian (it_IT), Japanese (ja_JP), Dutch (nl_NL), Swedish (sv_SE),
#     #   Brazilian Portuguese (pt_BR), Simplified Chinese (zh_CN),
#     #   Traditional Chinese (zh_TW), Danish (da_DK), Finnish (fi_FI),
#     #   Korean (ko_KR), Norwegian (nb_NO)
#     #
#     # You don't need to specify them for those languages; if you fail to
#     # specify them for some other language, English will be used instead.

#     'en_US': (
#         b'English',
#         b'Agree',
#         b'Disagree',
#         b'Print',
#         b'Save',
#         b'If you agree with the terms of this license, press "Agree" to '
#         b'install the software.  If you do not agree, press "Disagree".'
#     ),
# },
# }

```



## CHAPTER 2

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



## A

arrange\_by (built-in variable), 6

## B

background (built-in variable), 5

badge\_icon (built-in variable), 5

## D

default\_view (built-in variable), 6

## F

filename (built-in variable), 4

files (built-in variable), 4

format (built-in variable), 4

## G

grid\_offset (built-in variable), 6

grid\_spacing (built-in variable), 6

## I

icon (built-in variable), 5

icon\_locations (built-in variable), 5, 7

icon\_size (built-in variable), 7

include\_icon\_view\_settings (built-in variable), 6

include\_list\_view\_settings (built-in variable), 6

## L

label\_pos (built-in variable), 7

license (built-in variable), 8

list\_calculate\_all\_sizes (built-in variable), 7

list\_column\_sort\_directions (built-in variable), 8

list\_column\_widths (built-in variable), 8

list\_columns (built-in variable), 7

list\_icon\_size (built-in variable), 7

list\_scroll\_position (built-in variable), 7

list\_sort\_by (built-in variable), 7

list\_text\_size (built-in variable), 7

list\_use\_relative\_dates (built-in variable), 7

## S

scroll\_position (built-in variable), 7

show\_icon\_preview (built-in variable), 6

show\_pathbar (built-in variable), 5

show\_sidebar (built-in variable), 5

show\_status\_bar (built-in variable), 5

show\_tab\_view (built-in variable), 5

show\_toolbar (built-in variable), 5

sidebar\_width (built-in variable), 6

size (built-in variable), 4

symlinks (built-in variable), 5

## T

text\_size (built-in variable), 7

## V

volume\_name (built-in variable), 4

## W

window\_rect (built-in variable), 6