Preparing for a release under MacOSX

The following acts as a tutorial and a reference checklist for preparing your build for release and packaging for deployment. It covers a checklist of things for your config files, preparing and converting an icon for your executable, and finally how to create a disk image with an “installer”.

Code and Config pre-flight checklist

- Ensure that your [Resource] section has paths that will locate your config when you repackage your binary, configs, assets and library. For example:

```
Texture = ../../data # ./data
```

- Remove any test features from your code.
- Remove any files that aren't needed. Remove liborxd.dylib and liborxp.dylib. Only liborx.dylib is needed for release.

Preparing the application folder structure

Prepare up the following suggested folder structure, eg:

```
My Game Application
    Contents
        MacOS
        Resources
```

Under MacOS, drop in your executable, liborx.dylib, and starting ini file.

Ensure your executable is named the same as the root folder name, eg: “My Game Application” (without quotes) Also ensure your starting .ini is using the same name so the executable can pick it up, eg: “My Game Application.ini” (without quotes)

Under the Resources subfolder, drop in your data assets and any other ini files.

Make sure all config paths and resource paths are correct, by loading your executable and ensuring it all runs fine.

Icons

You will need to create an icns file, which is a file that can contain one or more icons of varying sizes. Get a PNG image that you would like to use.

Preparing the correct icon size

1. Double click your PNG to load into Preview.

2. Use Tools/Resize to change the size to 128×128 or 256×256 whatever is closest.
3. Export to a new PNG file and name using either the exact name of icon_128x128.png or icon_256x256.png

Building the icon folder structure to make a conversion

1. Create a folder called: icon.iconset
2. Copy your icon_128x128.png or icon_256x256 (or both) into this folder
3. Open a terminal and make your way to the parent folder of the icon.iconset folder
4. Type: iconutil -c icns icon.iconset

A new file called icon.icns will be created.

Rename this file in the same format of the application, eg: “My Game Application.icns” (without quotes).

Copy the file to Resources folder.

Creating package information

Package information is a way to add metadata into your application which identifies the name of the game, author, versions etc etc etc.

The Pkginfo file

Create a Pkginfo file under the Contents folder. Add the following literal text to the file:

```
APPL????
```

The Info.plist file

Create another file in the Contents folder called: Info.plist

Paste in the following and change the string values to what suits your application:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>BuildMachineOSBuild</key>
    <string>13A561</string>
    <key>CFBundleDevelopmentRegion</key>
```

```
    <string>English</string>
    <key>CFBundleExecutable</key>
    <string>My Game Application</string>
    <key>CFBundleGetInfoString</key>
    <string>1.0, Copyright © 2014-2015, Mr Developer.</string>
    <key>CFBundleHelpBookFolder</key>
    <string>My Game Application.help</string>
    <key>CFBundleHelpBookName</key>
    <string>net.somedomain.somesubdomain.help</string>
    <key>CFBundleIconFile</key>
    <string>My Game Application.icns</string>
    <key>CFBundleIdentifier</key>
    <string>net.somedomain.somesubdomain.somesubsubdomain</string>
    <key>CFBundleInfoDictionaryVersion</key>
    <string>1.0</string>
    <key>CFBundleName</key>
    <string>My Game Application</string>
    <key>CFBundlePackageType</key>
    <string>APPL</string>
    <key>CFBundleShortVersionString</key>
    <string>1.0</string>
    <key>CFBundleSignature</key>
    <string>????</string>
    <key>CFBundleVersion</key>
    <string>123</string>
    <key>DTCompiler</key>
    <string>net.apple.compilers.llvm.clang.1_0</string>
    <key>DTPlatformBuild</key>
    <string>5A11344p</string>
    <key>DTPlatformVersion</key>
    <string>GM</string>
    <key>DTSDKBuild</key>
    <string>13A561</string>
    <key>DTSDKName</key>
    <string></string>
    <key>DTXcode</key>
    <string>0500</string>
    <key>DTXcodeBuild</key>
    <string>5A11344p</string>
    <key>LSApplicationCategoryType</key>
    <string>public.app-category.arcade-games</string>
    <key>LSApplicationSecondaryCategoryType</key>
    <string>public.app-category.simulation-games</string>
    <key>LSHasLocalizedDisplayName</key>
    <true/>
    <key>LSMinimumSystemVersion</key>
    <string>10.8.0</string>
    <key>NSMainNibFile</key>
    <string>My Game Application</string>
    <key>NSPrincipalClass</key>
    <string>NSApplication</string>
```

```
    <key>NSSupportsSuddenTermination</key>
    <true/>
</dict>
</plist>
```

Here is a handy guide to cover the keys above:
https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Articles/CoreFoundationKeys.html#//apple_ref/doc/uid/TP40009249-SW1

And wikipedia will help you locate values for your `BuildMachineOSBuild` value. Start at:
https://en.wikipedia.org/wiki/MacOS#Release_history and look under the “Most Recent Version” column to get a version code.

The version.plist file

Create another file under the Contents folder called: version.plist. Paste in the following and change the string values to what suits your application:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>BuildVersion</key>
    <string>1</string>
    <key>CFBundleShortVersionString</key>
    <string>1.0</string>
    <key>CFBundleVersion</key>
    <string>1.0</string>
    <key>ProjectName</key>
    <string>My Game Application</string>
    <key>SourceVersion</key>
    <string>1</string>
</dict>
</plist>
```

Converting structure to an application

Rename the “My Game Application” to “My Game Application.app”

Your icon should appear on the application and it will appear as a single file.

Double click on the Application file to check that the application launches fine.

Creating a disk Image

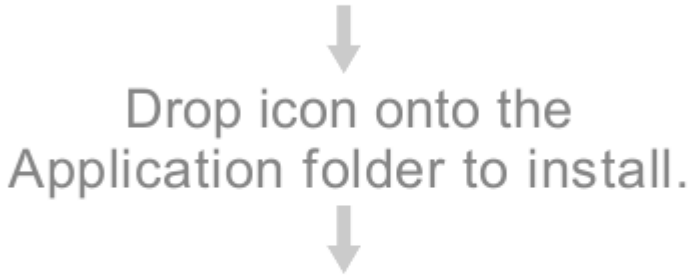
In order to distribute your game, you'll need to turn the “My Game Application.app” folder structure into a disk image.

1. Create a folder with the same name as your app, e.g. “My Game Application”, and drag your .app file into it,
2. Launch the Disk Utility application.
3. Select File / New / Image from Folder
4. Select your “My Game Application” folder and click the Open button.
5. Save as “My Game Application.dmg” and choose read/write as the image format.
6. Click Save.
7. Your .dmg file will be created.
8. Close Disk Utility
9. Double click on the DMG file to mount it as a Volume.
10. Double click the “My Game Application” volume to open it. You will see your game application icon inside.
11. Double click your game icon to check it all starts up ok.

Creating an Installation

Rather than an empty window with your icon, you can change this window to be more like an installer, giving the user instructions on how to install your game to onto the player's Mac.

Start with a background image like the following:



Save this image as: mac-installer-background.png.

Setting an installer background

1. Open the “My Game Application” volume again if it isn't already.
2. Create a folder titled “background”
3. Drag the mac-installer-background.png into the “background” folder.
4. Return to the main DMG window where your game icon is.
5. Select View / Show View Options
6. Change the “Background” radio option to “Picture”
7. Drag the mac-installer-background.png file in the “background” folder onto the “Drag Image Here” box.
8. The DMG window should have its background changed to the mac-installer-background.png image.

Hiding the background folder

1. To hide the background folder, go to the terminal and make your way to the “My Game Application” volume. (usually with cd /Volumes/My Game Application)
2. Enter: SetFile -a V background (this will hide the folder)

Placing icons for dragging the game to the Application folder

1. Drag your game icon into a good position at the top of the window.
2. Open another Finder window and go to the Macintosh HD.
3. Right click the “Applications” folder and select “Make Alias” to make a shortcut.
4. Drag this shortcut to the DMG window. Rename it to remove “alias” from the name and drag it to a nice position at the bottom of the window.
5. Resize the window itself to a nice suitable size.

Close the terminal, and any open windows to the “My Game Application” volume and then Eject it.

Compressing the image

Finally, you need to compress the image to make it smaller, and no longer writable.

1. Open the `Disk Utility` application.
2. Select Images / Convert.
3. Select “My Game Application.dmg”. Click “Open”.
4. Save to a new .dmg file elsewhere.
5. Ensure Image Format is set to “compressed” and select “Save”

The result is a compressed “My Game Application.dmg” disk image that can now be mounted and tested. A user can download this file, drag the game application into the Applications folder, and then throw the DMG away.

Additional Tips

There are a number of other things you might like to consider when packaging your release.

Troubleshooting

Icons: when trying to make the icon.ics file, you may receive the following:

```
'icon' resource error: 'Failed to load image at 16@1x.'
```

If so, you can probably ignore these warnings. Check that `icon.ics` file was created anyway, and contains the icons you specified.

Application error: When running your *.app file, and you receive a: `You can't open the application because it may be damaged or incomplete.`

Ensure that your *.app folder name matches your key name which matches your actual executable filename.

Converting DMG: If you receive an error during Disk Utility converting, ensure that the original dmg is unmounted.

From:
https://www.orx-project.org/wiki/ - **Orx Learning**

Permanent link:
https://www.orx-project.org/wiki/en/tutorials/publishing/preparing_a_macosx_release

Last update: **2025/09/30 17:26 (14 months ago)**