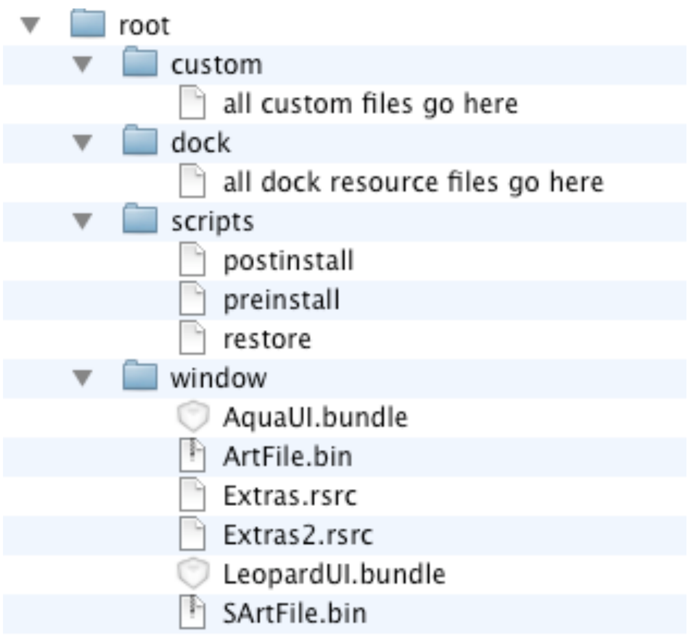
Magnifique Developer's Guide

So, you want to develop a theme. Developers are important, Magnifique would not be here without them. We've made the theme making process as easy as possible. To create basic theme files you do not need to know any coding whatsoever. Here's how to start.

The only real part the developer needs to do is create a "root" folder. The root folder contains everything the theme will install. Here's a basic tree map of what a root folder CAN contain (doesn't have to contain everything, but this is what its possible to contain):

```
root
        -- window
                --- ArtFile.bin
                --- AquaUI.bundle
                --- Extras.rsrc
                --- Extras2.rsrc
                --- LeopardUI.bundle
                --- SArtFile.bin
        -- dock
                --- <any dock resource file>
        -- custom
                --- <any custom files organized by directory structure>
        -- scripts
                --- preinstall
                --- postinstall
                --- restore
```

Here's a real Finder picture of what this would look like:

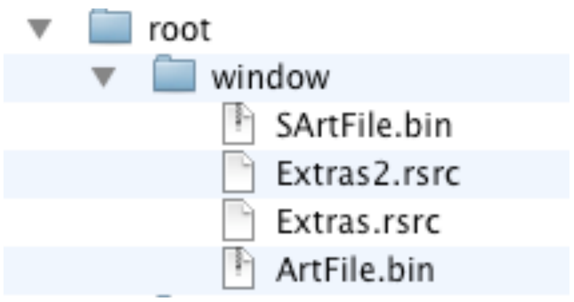


```
▼ root
    ▼ custom
        all custom files go here
    ▼ dock
        all dock resource files go here
    ▼ scripts
        postinstall
        preinstall
        restore
    ▼ window
        AquaUI.bundle
        ArtFile.bin
        Extras.rsrc
        Extras2.rsrc
        LeopardUI.bundle
        SArtFile.bin
```

The “window” folder = Main system (window theme), “dock” folder = Dock theme and “custom” and “scripts” folders go together, they = custom theme. You can have one or any combination of these 3 to use (again, not all the files have to be there, only the ones you will use). **Note that each of these 3 will be presented as options on the “Apply Theme” dialog, and they all won’t just automatically installed (you can pick and choose Dock, main system-window theme, and custom mods, choosing one, two, or all three)** Let’s go through each of the 3 in order.

Window (main system theme).

This has the ability to install ArtFile.bin, SArtFile,bin, Extras.rsrc, Extras2.rsrc, LeopardUI.bundle, AquaUI.bundle. You don’t need to include all the files, only the ones you modified. For example here is the root >> window folder of the iLeopard theme:

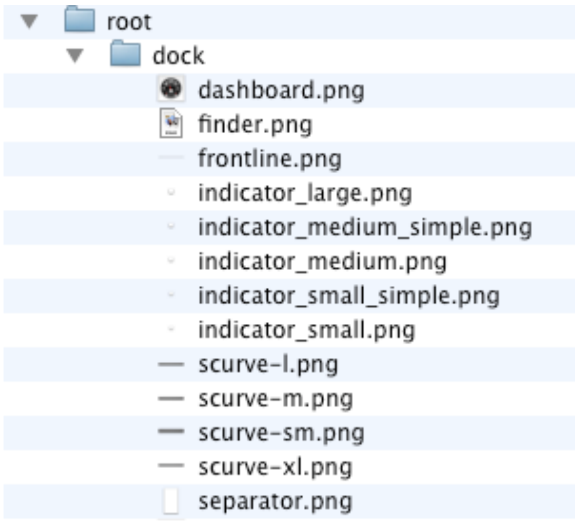


```
root
  window
    SArtFile.bin
    Extras2.rsrc
    Extras.rsrc
    ArtFile.bin
```

Just like it looks, create a “window” folder inside the root folder and put the files in there. Note that this “window” folder will ONLY install the files mentioned above, it will not install anything else.

Dock theme

This adds a Dock theme. Simply create a “dock” folder inside the root folder and put your modified Dock resource files in there. Anything in the “dock” folder will be installed to /System/Library/CoreServices/Dock.app/Contents/Resources/ overwriting any existing files. Here is a screenshot of some of the files in a dock theme:



```
root
  dock
    dashboard.png
    finder.png
    frontline.png
    indicator_large.png
    indicator_medium_simple.png
    indicator_medium.png
    indicator_small_simple.png
    indicator_small.png
    scurve-l.png
    scurve-m.png
    scurve-sm.png
    scurve-xl.png
    separator.png
```

Custom theme

The most flexible, and also the hardest one to do. This allows you to install ANY file to the system. 2 sections for this, placing the files, and writing scripts.

-- Placing files

For custom, files are placed in a hierarchal directory structure. For example, lets say you had the custom file “/System/Library/CustomFile.X”. To do this you would use a structure like this:

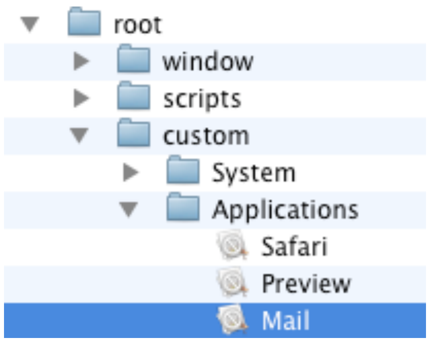
```
root
        -- custom
                --- System
                        ---- Library
                                ---- CustomFile.X
```

So basically you create the “custom” folder inside root. Then you create a System folder inside custom, then a Library folder inside System, and then put CustomFile.X in the Library folder. So in this method you can place as many files as you want using the custom structure.

Now, wait. What if you want to make a modification to an app (.app), or some sort of plugin (.plugin) or bundle (.bundle), or anything that is basically a folder that shows up as a single file. Do you have to copy the entire thing just for a few mods? No. Say you have Mail.app (**or any app, we’ll use Mail.app as an example**) and you just want to put a file called “customicon.x” in /Applications/Mail.app/Contents/Resources. Just do this:

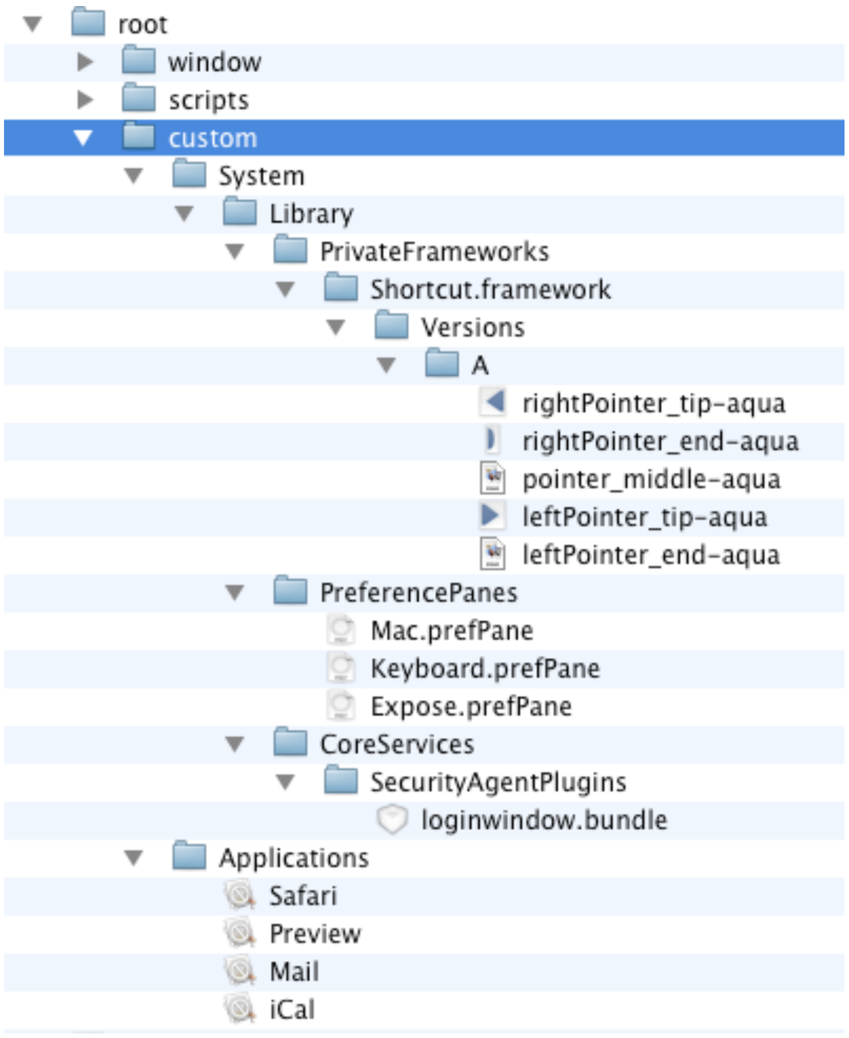
```
root
        -- custom
                --- Applications
                        ---- Mail.app
                                ----- Contents
                                        ----- Resources
                                                ------- customicon.x
```

Basically create an empty folder called “Mail.app” inside Applications in the custom folder. Mail.app will automatically turn into an app file with a cross over it like this:



Then right click Mail.app >> Show Package Contents. Then inside Mail.app create the Contents folder, then inside that, create the Resources folder, and put your "customicon.x" or whatever file in there. This same procedure can be used for bundles, plugins, packages, anything that you can "Show Package Contents" on.

That should wrap up file placement on custom themes. Here is an example of the file structure of iLeopard's custom folder:



```
▼ root
  ▶ window
  ▶ scripts
  ▼ custom
    ▼ System
      ▼ Library
        ▼ PrivateFrameworks
          ▼ Shortcut.framework
            ▼ Versions
              ▼ A
                  rightPointer_tip-aqua
                  rightPointer_end-aqua
                  pointer_middle-aqua
                  leftPointer_tip-aqua
                  leftPointer_end-aqua
        ▼ PreferencePanes
            Mac.prefPane
            Keyboard.prefPane
            Expose.prefPane
        ▼ CoreServices
          ▼ SecurityAgentPlugins
              loginwindow.bundle
    ▼ Applications
        Safari
        Preview
        Mail
        iCal
```

When a custom theme is installed, the "preinstall" script will run (if there is one), then the contents of the custom folder will be cloned onto the root drive, and then the postinstall script will run (if there is one). What are these scripts? That's what the next section is about

-- Writing Scripts

Because of the vast number of items a custom script can install, Magnifique requires that developers write their own scripts if they are using the custom feature. There are 3 scripts that can be used. **preinstall**, **postinstall**, and **restore**. These are placed into root/scripts/ and are run when a custom mod is installed, preinstall before copying the custom files, postinstall after the copy, and restore runs when a custom mod is uninstalled. **These are ONLY used for "custom"**. preinstall and restore are the only ones that every custom mod should have, postinstall is just optional. **Scripts are placed in root/scripts**

These script files must have NO extension (no .txt or anything) and must be named exactly as above. TextMate is an excellent application for writing scripts, just make sure that the files are encoded in a plain text format (no rich text) and if they have a file extension, remove it (Right click file in Finder >> Get info >> Name and Extension-- make sure to uncheck the "Hide extension" box, then chop off the extension at the end). **The scripts are written in shell/bash script, so make sure you have a knowledge of that (basically just Terminal commands, pasted one per line). You do not need to worry about "sudo" as every script is run with admin privileges.**

--- Preinstall script

Here is an example of a preinstall script for the iLeopard theme (commented to show what does what).

#########SCRIPT START########

```
#!/bin/sh

# Removes any existing iLeopard Backup folder
rm -rf "/Library/Application Support/Magnifique/Backups-Custom/iLeopard"
# Makes a new iLeopard backup folder
mkdir "/Library/Application Support/Magnifique/Backups-Custom/iLeopard"
# Copies Safari.app to the iLeopard Backup folder (cp with the -R flag must
be used when copying folders, apps, packages, bundles, plugins, etc.)
cp -R /Applications/Safari.app "/Library/Application Support/Magnifique/
Backups-Custom/iLeopard/"
# Copies loginwindow.bundle to the backup folder, and so on to copy the rest
of the files....
cp -R /System/Library/CoreServices/SecurityAgentPlugins/loginwindow.bundle "/
Library/Application Support/Magnifique/Backups-Custom/iLeopard/"
cp -R /System/Library/PreferencePanes/Mac.prefPane "/Library/Application
Support/Magnifique/Backups-Custom/iLeopard/"
cp -R /System/Library/PreferencePanes/Keyboard.prefPane "/Library/Application
Support/Magnifique/Backups-Custom/iLeopard/"
cp -R /System/Library/PreferencePanes/Expose.prefPane "/Library/Application
Support/Magnifique/Backups-Custom/iLeopard/"
cp -R /System/Library/PrivateFrameworks/Shortcut.framework "/Library/
Application Support/Magnifique/Backups-Custom/iLeopard/"
```

```
cp -R /Applications/Mail.app "/Library/Application Support/Magnifique/
Backups-Custom/iLeopard/"
cp -R /Applications/iCal.app "/Library/Application Support/Magnifique/
Backups-Custom/iLeopard/"
cp -R /Applications/Preview.app "/Library/Application Support/Magnifique/
Backups-Custom/iLeopard/"
```

#########SCRIPT END########

Basically what the preinstall script does is it backups up anything that will be overwritten by the app. **Its best to backup the entire app/framework/bundle whatever rather than just the Resources folder. As you can see, every script starts with "#!/bin/sh" to tell the computer to use the shell interpreter to run the script.** The designated folder for Custom Backups is /Library/Application Support/Magnifique/Backups-Custom/<your theme name> . Name your backup folder something unique so other themes won't overwrite it. Make sure to delete any existing backup folders for your theme by:

rm -rf "/Library/Application Support/Magnifique/Backups-Custom/<your backup folder>"

Then create the backup folder again using:

mkdir "/Library/Application Support/Magnifique/Backups-Custom/<your backup folder>"

Also note that when using a file path with spaces in it, put QUOTES (" ") around the path. To copy files and folders to the backup folder just use:

cp -R <file/folder to copy> "/Library/Application Support/Magnifique/Backups-Custom/<your backup folder>/"

Example for iLeopard:

cp -R /Applications/Safari.app "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/"

That's about it for the preinstall script! Postinstall script isn't required unless you want to do post installation actions, but its the same shell script concept (you can figure that out yourself), so onto the restore script.

-- restore script

The restore script is what runs when a theme is uninstalled. It basically restores all the original files that were backed up in the preinstall script. PLEASE include a restore script in your theme if you are installing custom files. Without a restore script its extremely HARD for end users to revert back changes. Here is an example of a restore script from iLeopard:

#########SCRIPT START########

```
#!/bin/sh

# Copies the backup Safari.app to Applications/Safari.app. The "ditto"
command basically clones a file/folder onto another destination
ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/
Safari.app" /Applications/Safari.app
# Dittos loginwindow.bundle to /System/Library/CoreServices/
SecurityAgentPlugins/loginwindow.bundle and so on for the rest of the
files...
ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/
loginwindow.bundle" /System/Library/CoreServices/SecurityAgentPlugins/
loginwindow.bundle
ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/
Mac.prefPane" /System/Library/PreferencePanes/Mac.prefPane
ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/
Keyboard.prefPane" /System/Library/PreferencePanes/Keyboard.prefPane
ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/
Expose.prefPane" /System/Library/PreferencePanes/Expose.prefPane
ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/
Shortcut.framework" /System/Library/PrivateFrameworks/Shortcut.framework
ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/
Mail.app" /Applications/Mail.app
ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/
iCal.app" /Applications/iCal.app
ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/
Preview.app" /Applications/Preview.app
# Deletes the backup folder once its done. We don't want to fill up the HD
rm -rf "/Library/Application Support/Magnifique/Backups-Custom/iLeopard"
```

#########SCRIPT END########

The function is pretty basic. Restores the backup of a file to the system using the ditto command:

ditto <your backup file> <destination location including the name of the file>

Example:

ditto "/Library/Application Support/Magnifique/Backups-Custom/iLeopard/ Safari.app" /Applications/Safari.app

Make sure that you put the full destination path to the app/file so not just "/Applications" but "/Applications/Safari.app" Now at the end of the restore script, we just delete the backup folder to save some free space:

rm -rf <backup folder>

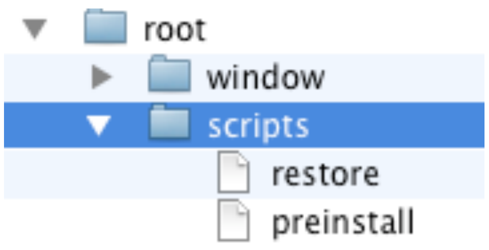
Example:

rm -rf "/Library/Application Support/Magnifique/Backups-Custom/iLeopard"

That just about wraps up the scripts section! **Remember you can include a "postinstall" script too** if you would like, but its not required and its only if you want to do post installation actions AFTER the custom files are installed.

Put all scripts into root folder >> scripts

Here is what a scripts folder looks like:



==========================

Mix n' Match Method

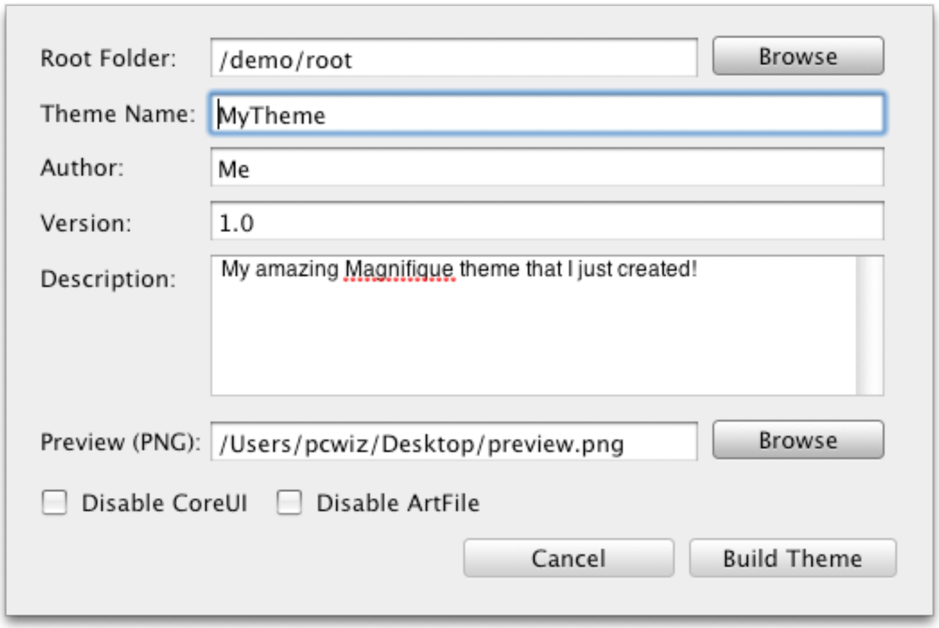
Using the mix n' match method you can combine elements of different themes to create your own customized look. This is useful to say, if you want to use the window (main system) theme from one theme and a dock from the other. To start, create an empty "root" folder somewhere.

Now go into /Library/Application Support/Magnifique/Themes to find your theme library. In there right click the theme you want to take contents from and click Show Package contents. Go into the root folder of the theme and you will see one or more of these folders: window, dock, custom, and scripts. If you want the window (main system) theme, copy the window folder to your empty root folder. **Also if you are copying the window folder and see file(s) called "discoreui" and "disartfile" in the root folder of the source theme then copy those as well**. If you want the dock, then copy the dock folder to your empty root folder. If you want the custom mods, copy BOTH the custom AND the scripts folders to your empty root folder. This way you can pick and mix contents from different themes (window folder from one theme, dock folder from the other, etc. Once you have your root folder all ready to go, you can use the same theme compiling method (below) that developers use for their own themes

==========================

Compiling your theme

So you have your root folder all ready to go, with your dock, custom + scripts, and window folders (or a combination of them). There is only one last step before making the package. Take a preview picture of your theme (its optional, but gives much better first impressions)! Maybe its a picture of your OS X desktop with the theme, or a Finder window, or a collage with the different parts of the theme. The only requirement is that the picture must be in PNG format. Once you got your root folder and picture ready to go, open Magnifique and click Make Theme. You get a dialog like this:



Root Folder:	/demo/root
Theme Name:	MyTheme
Author:	Me
Version:	1.0
Description:	My amazing Magnifique theme that I just created!
Preview (PNG):	/Users/pcwiz/Desktop/preview.png

Choose your root folder, give your theme a name, and enter details for Author and Version. For the description, enter any additional information, a summary of your theme, what it contains, etc. etc. Then select your preview PNG file. There are 2 options at the bottom for disabling CoreUI and ArtFile. If your theme requires that CoreUI or ArtFile be disabled, then check the appropriate checkbox. Once you are done, click Build Theme. It will copy and set everything up for you into a nice .mfq.plugin file. Your theme will be saved in /Library/Application Support/Magnifique/Themes/ so if you want to distribute it, copy it from there.

Testing

The final (and most important) step before distributing your theme to the public. The last thing you want is a theme that messes up peoples' computers, or one that just does not work. Preferably using a spare OS X install (or your existing one, backup all important data first in case the experiment goes wrong!!!), test applying the theme and uninstalling as well (to test the restore script). Make sure that everything works as it should, and that there are no side effects/errors or issues.

Distribution

Once everything is tested and working, release it out to the public! Let everyone enjoy your creation. Hopefully this guide has been helpful on your way to developing a theme for Magnifique, the free theme manager. **Note: If you used elements from other themes, make sure to get permission from the original authors and give credit.**