Create your own pacman package for Batocera

What is pacman?

In Batocera **v27** and higher, you can utilize pacman to install and manage packages. It is the underlying tool behind the content downloader. It is a straightforward package manager that's easy to work with.

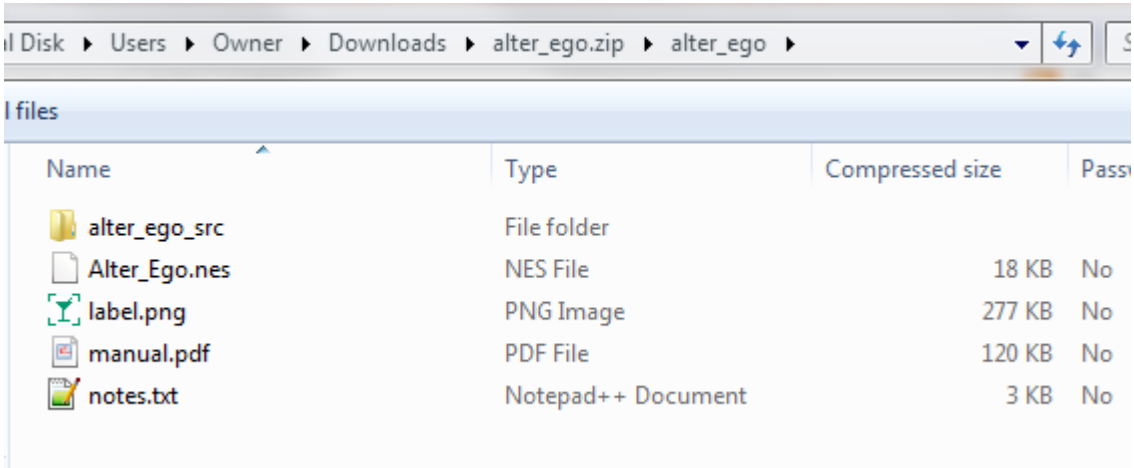
You don't need development skills to create a package, just a bit of patience. If you've copied files too and from directories before, you already have the skill-set required for this.

Create your own package

Pre-setup

The way packages work is that they are essentially just a compressed folder that contains the folder structure that you wish to copy onto the user's machine, starting at root, along with a script file. Essentially, you'll be creating a “userdata” folder inside of the beginning folder of your package, and go from there.

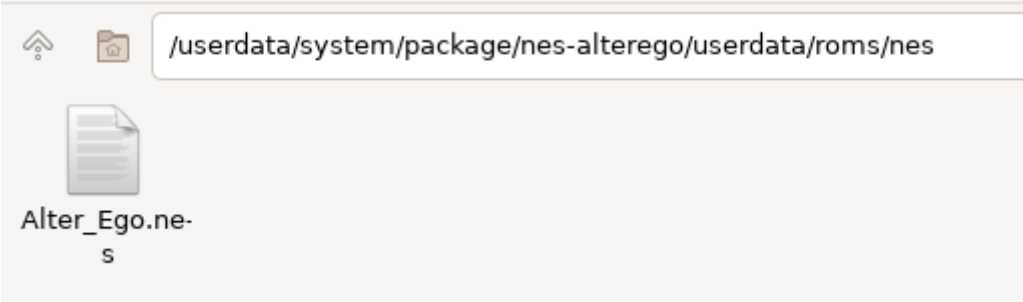
It's easiest to explain with an example. Let's use the NES free homebrew ROM Alter Ego available from https://www.romhacking.net/homebrew/1/. Click “Download” from the **Links** section and fill out the captcha to get the ZIP and open it up, it will contain the following contents:



This contains everything we might need to create a package including media and all!

For the following steps, it is recommended to use Batocera's file manager (`[F1]` on the system list). You can transfer the files over to Batocera first using one of the many methods available.

1. On your Batocera machine, create a new folder in `/userdata/system/` and call it `package/`. This folder can technically be anywhere but this is what we will be using for our example.
2. Inside of `package/`, create another folder named after your package (in this case, `nes-alter-ego/`).
3. Inside of the package's folder, recreate the folder structure for every relevant file that needs to be copied over to the user's machine.
4. Place the relevant files into their respective folders. For instance, if all you want to add is a ROM for the NES system, then you would simply create the `userdata/roms/nes/` folders inside of each other in the package's folder and place the ROM in the final `nes/` folder:



In our example package, which includes the media we want to install, the setup would look like this:

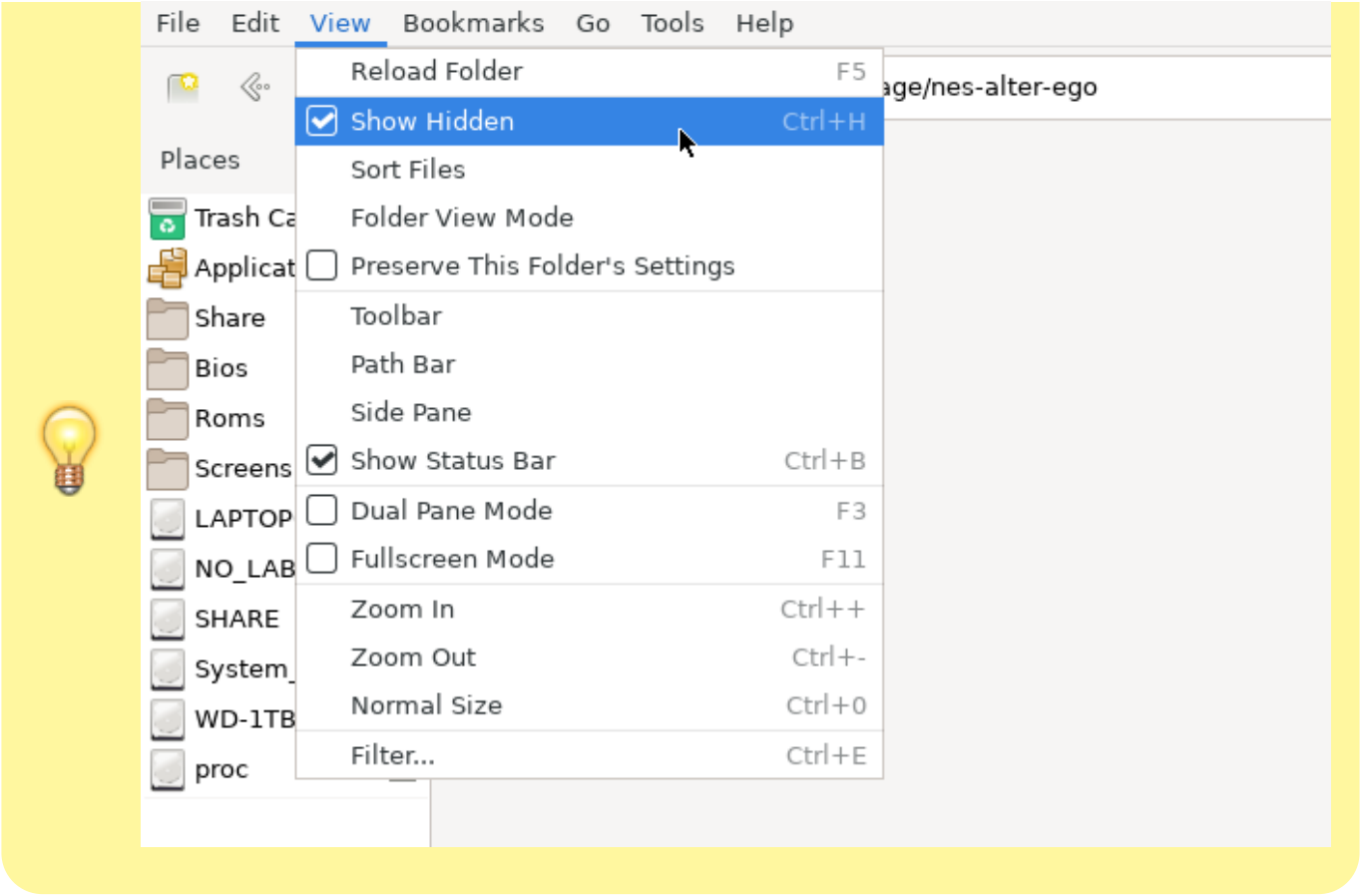
```
/userdata/system/package/nes-alter-ego/
                          └─ userdata/
                             └─ roms/
                                └─ nes/
                                   ├─ images/
                                   │  └─ label.png
                                   ├─ manuals/
                                   │  └─ manual.pdf
                                   └─ Alter_Ego.nes
```

However, these filenames simply won't do! Batocera won't be able to recognize the media files as belonging to Alter Ego. Let's rename them:

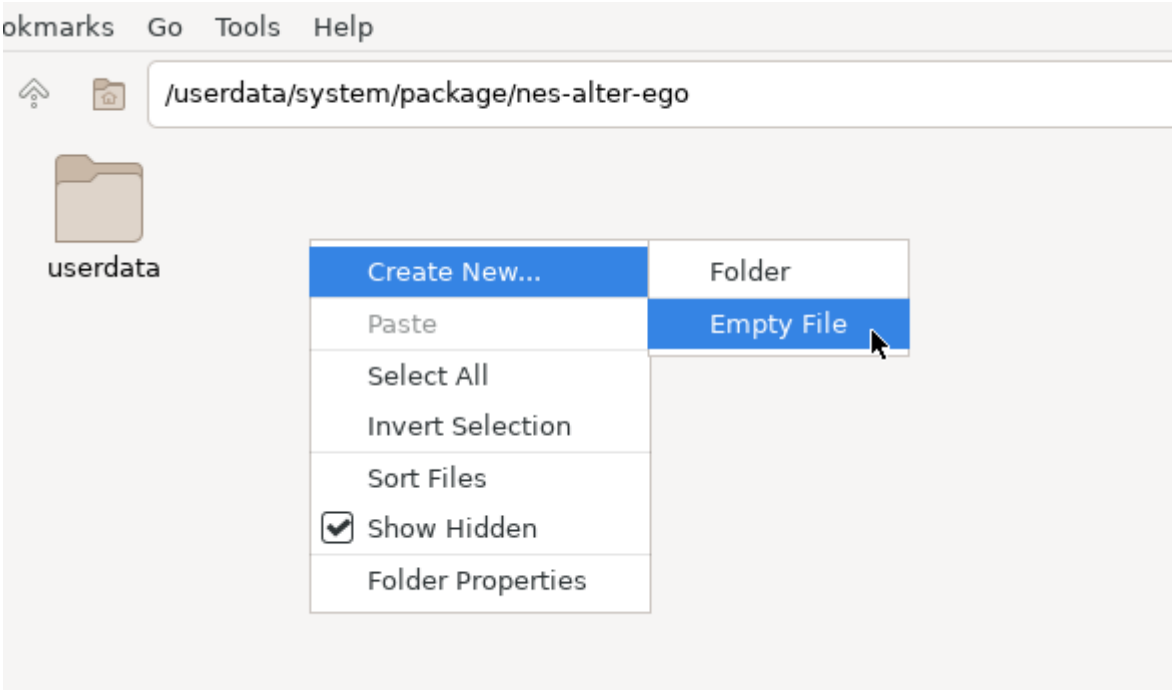
```
/userdata/system/package/nes-alter-ego/
                          └─ userdata/
                             └─ roms/
                                └─ nes/
                                   ├─ images/
                                   │  └─ Alter_Ego.png
                                   ├─ manuals/
                                   │  └─ Alter_Ego-manual.pdf
                                   └─ Alter_Ego.nes
```

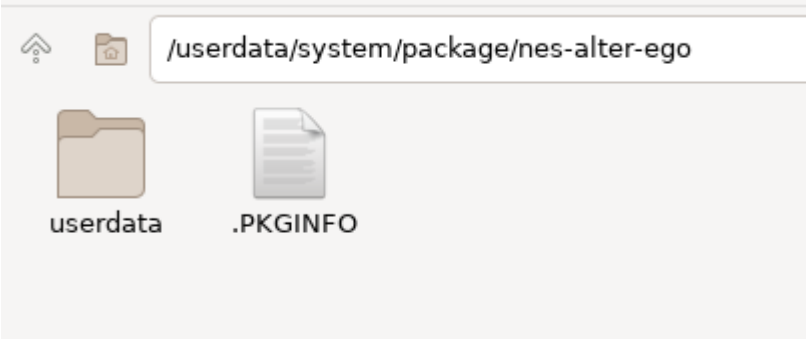
Package info file

In order to see hidden files in Batocera, ensure that “Show Hidden” is checked in the **View** menu at the top left.



Next we need to create a hidden file that contains the meta-information about the package itself. Create a new text file named `.PKGINFO` in the package's folder. For our example, this would be `/userdata/system/packaging/nes-alter-ego/.PKGINFO`.





This file must be named `.PKGINFO`. If unable to create a file with that name for whatever reason, you can download the script below and place it in `system/packaging/nes-alter-ego/` on the network share (you may need to turn on “Show hidden files/folders” in your file manager).

.PKGINFO

```
pkgname = nes-alter-ego
pkgver = 1.0.0-1
pkgdesc = Alter Ego - A freeware puzzle game by Shiru, Kulor, Denis
Grachev.
arch = any
group = sys-nes
packager = your_name
url = https://wiki.batocera.org/start
```

Insert the lines above into the `.PKGINFO` text file (you can open it by just double-clicking). Replace the lines as appropriate! Here are the **required** lines:

- **pkgname:** The internal name of the package. This needs to remain the same for updates that become available for the package.
- **pkgver:** The version of the package. If this number is higher than the one currently installed, Batocera will offer the user to update the package in the store. The only acceptable format for this is `X.Y.Z-R`, following pacman's conventions.
- **pkgdesc:** The package's displayed name, following by a short description. This will be what appears in the content downloader. You only have one line of text, around 80 characters, so keep it brief!
- **arch:** What platforms should this package be available to? Most packages are fine to leave this as “any”, but some may only work on x86_64 for example.
- **group:** What category should the store show this under? If you are creating a package for a specific system, it must be prefixed with `sys-`, eg. `sys-megadrive` for Megadrive/Genesis. For the other groups, read below.

The remaining lines are optional, but can provide extra information to users choosing to install them:

- **packager:** The author of the package.
- **url:** URL for the user to go to if they desire more information. For freeware, generally links to the page the download can be made manually.

Ensure there is a blank line at the bottom of the text file. This is critical.

```
File  Edit  Search  Options  Help
pkgname = nes-alter-ego
pkgver = 1.0.0-1
pkgdesc = Alter Ego - A freeware puzzle game by Shiru, Kulor, Denis Grachev.
arch = any
group = sys-nes
packager = your_name
url = https://wiki.batocera.org/start
```

Available groups

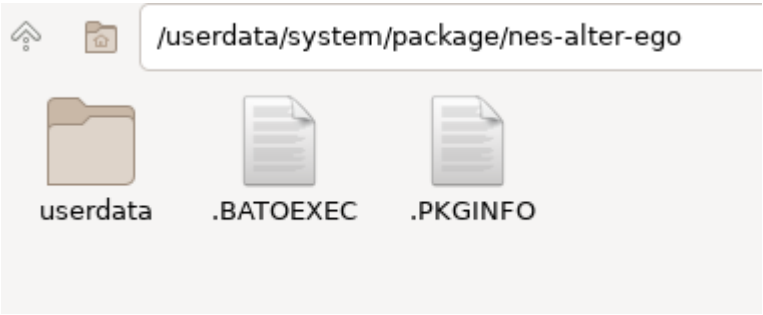
In addition to the `sys-<system>` groups that are available, you have the following to choose from:

- **music:** Add some more bangin' tunes.
- **theme:** Add an additional theme for the menu.
- **bezel:** Various decorations!
- **misc:** All the other junk.

Additional actions

If you are simply copying files to the machine, you can move onto creating the package. However, if you need to run some commands (such as to edit already existing files, such as `gamelist.xml`), read this section.

The other file is `.BATOEXEC` and it gives you the ability add metadata or execute commands. The first line describes the action you want to do. Currently, BATOEXEC accepts two commands: `gamelist` and `exec`. Create a new blank document name `.BATOEXEC` and place it in the same folder next to `.PKGINFO`.



gamelist

Typically, for a package that adds a new ROM, you want to add it into the `gamelist.xml`. Add `gamelist = <system>`, where `<system>` is the shortname of the system you'd like to edit the `gamelist.xml` for, and then follow it with the data you'd like to append to the list. For example:

.BATOEXEC

```
gamelist = nes
<game>
  <path>./Alter_Ego.nes</path>
  <name>Alter Ego</name>
  <desc>Freeware puzzle game for NES, a port of the original (by Denis
Grachev for the ZX Spectrum). Swap positions with your 'alter ego' to
move about the level and obtain all the bouncing pixels.</desc>
  <rating>0.6</rating>
  <releasedate>20110827T000000</releasedate>
  <developer>Shiru</developer>
  <publisher></publisher>
  <genre>Puzzle</genre>
  <players>1</players>
  <image>./images/Alter_Ego.png</image>
  <manual>./manuals/Alter_Ego-manual.pdf</manual>
</game>
```

The `gamelist = <system shortname>` here is important. Don't forget to edit it appropriately for your system in question!

exec

This is for any other command you might want to run when installing the package. Its syntax depends on which executable you call in `exec = <bin>`.

Only interpreters that takes a `-c` argument for commands are available. For example, `exec = /bin/bash` or `exec = /usr/bin/python`, which are both used heavily by Batocera for its internal functions.

For a basic example that will be executed every time the package is called (whether you are installing, upgrading or uninstalling it):

.BATOEXEC

```
exec = /bin/bash
echo "You have successfully run a command in a script of a package!"
```

If you define sections between `.INSTALL_START` and `.INSTALL_END` or `.UNINSTALL_START` and `.UNINSTALL_END`, those commands will be executed only on install/upgrade or removal of the package respectively. Commands outside those sections will be executed in any case (ex: `date` in the

example below).

.BATOEXEC

```
exec = /bin/bash
date
.INSTALL_START
echo "Installing bezels with glazed effect..."
.INSTALL_END
.UNINSTALL_START
echo "Uninstalling bezels with glazed effect..."
.UNINSTALL_END
```

A `.BATOEXEC` file is not mandatory to use `batocera-makepkg`. Many installable objects don't require it, such as adding themes, music, etc.

Setup complete

This is what the package setup should look like:

```
/userdata/system/package/nes-alter-ego/
                         ├─ userdata/
                         │  └─ roms/
                         │     └─ nes/
                         │        ├─ images/
                         │        │  └─ Alter_Ego.png
                         │        ├─ manuals/
                         │        │  └─ Alter_Ego-manual.pdf
                         │        └─ Alter_Ego.nes
                         ├─ .PKGINFO
                         └─ .BATOEXEC
```

Create the package

Batocera uses a custom tool to create its packages: `batocera-makepkg`.

To use it:

1. Access Batocera via SSH.
2. Change the current working directory to the package's folder. In our example, this can be achieved with `cd /userdata/system/package/nes-alterego`.
3. Run `batocera-makepkg` to create the package. It will output something similar to the following:

```
Creating package ../nes-alter-ego-1.0.0-1-any.pkg.tar.zst ...
/*stdin*\            : 45.08%   ( 46592 =>  21005 bytes, ../nes-alter-
ego-1.0.0-1-any.pkg.tar.zst)
```

```
SUCCESS: package ../nes-alter-ego-1.0.0-1-any.pkg.tar.zst correctly
generated
```

This may take a while depending on the size of your package's contents.

When complete, a package file named `<system>-<pkgname>-<pkgver>-<arch>.pkg.tar.zst` will be created one level above your current working directory. In this example, it would be `/userdata/system/package/nes-alter-ego-1.0.0-1-any.pkg.tar.zst`

`batocera-makepkg` will not overwrite already existing packages with the same name.

`batocera-makepkg` will generate some necessary files in the package's source folder, such as if you had used `.BATOEXEC` it will create the `userdata/system/pacman/batoexec/nes-alter-ego_0` file which is a copy of the `.BATOEXEC`. These files can be completely ignored, as they will be updated with any edits made to the source files should any new version of the package be made.

Installing your package

If you'd like to test installing your package, you can do so with the following command:

```
pacman -U <package>
```

If you were following the example above, the command for it would be:

```
pacman -U /userdata/system/package/nes-alter-ego-1.0.0-1-any.pkg.tar.zst
```

Updating your package

If you kept the original package files, it is simple: make your adjustments and increment the version number however you see fit.

.PKGINFO

```
pkgname = nes-alter-ego
pkgver = 1.0.1-1
pkgdesc = Alter Ego - A freeware puzzle game by Shiru, Kulor, Denis
Grachev.
arch = any
group = sys-nes
packager = your_name
url = https://wiki.batocera.org/start
```

The `.PKGINFO` will now contain the `size` and `builddate` tags. These are updated automatically upon compilation, so don't worry about them.

You may notice that in your `userdata/` folder for the package, there are additional folders and files that you had not placed there yourself. These have been automatically generated by `batocera-makepkg` (such as if you had used `.BATOEXEC` it will create the `userdata/system/pacman/batoexec/nes-alter-ego_0` file which is a copy of the `.BATOEXEC`) and can be safely ignored; they will be updated along with the rest of your package.

When ready, follow the instructions from the Create the package section above. If you receive an error that the package already exists, that means you forgot to increase the package version number or that you have already compiled it.

When updating packages, you **must** increment the version number, even for minor changes. Otherwise, it may cause corruption if the user has the same numbered package installed upon attempting to update to a newer version.

Remove your package

When you're done testing or just want a clean slate to work with. Do note however, a ZST file is not actually required to remove a package. Simply enter the name (`pkgname`) of the installed package in the following command:

```
pacman -R <pkgname>
```

This is version independent.

For example:

```
pacman -R nes-alter-ego
```

which would result in:

```
checking dependencies...

Packages (1) nes-alter-ego-1.0.0-1

Total Removed Size:  1.33 MiB

:: Do you want to remove these packages? [Y/n]
```

Press `[Y]` followed by `[Enter]` to complete the removal.

To see a list of installed packages and their version numbers, run `pacman -Q`.

Troubleshooting

Why is my package so huge?

Most likely you didn't change the current working directory to the folder of the package you wanted to create. By default, the current working directory will be in `/userdata/system`, so all the folders and files in there will be added to your package.

Run `cd /userdata/system/package/nes-alter-ego` if you were following the example above and try again.

How can I see what's in the package?

The simplest answer is to install it and see what files were added. But that's not always easy, especially when there are multiple files in deep-rooted folder levels. If you'd like to "extract" a package file, you can use the `zstd` tool in Batocera:

```
zstd -d /userdata/system/package/nes-alter-ego-1.0.0-1-any.pkg.tar.zst
```

This will give you a standard `.tar` file that you can use any ordinary compression program to decompress.