packmaker Documentation

Mark Crewson

Jul 02, 2020

Contents

Packmaker is a command line tool used to build Modded Minecraft modpacks on Linux systems. It can take a yaml-based description of a pack and generate a package suitable for use in the following formats:

- **Curseforge** - A curseforge package suitable for uploading directly to the curseforge.com webiste, or importing into a curseforge compatible minecraft launcher like MultiMC or the Twitch Client.
- **Routhio** - A package format used by the minecraft.routh.io community and the routh.io minecraft launcher.
- **Local client** - A local minecraft client installation, suitable for standalone games. Packmaker can even launch your modded minecraft installation in offline mode itself. Great for testing your modpack during development.
- **Server** - A minecraft server installation, suitable for use by clients to connect with and play together.

If you're new to packmaker, begin with the *Getting Started* guide. That guide walks you through installing packmaker, setting it up how you like it, and starting to build your first modpack.

Then you can get a more detailed look at packmaker's features in the *Command-Line Interface* and *Configuration* references.

If you still need help, you can drop by the `#packmaker` channel on our Discord server, or file a bug in the issue tracker. Please let us know where you think this documentation can be improved.

And if you're really ambitous, checkout the code and help us improve packmaker for you and everyone.

CHAPTER 1

Contents

1.1 Guides

This section contains a couple of walkthroughs that will help you get familiar with packmaker. If you're new to packmaker, you'll want to begin with the *Getting Started* guide.

1.1.1 Getting Started

Welcome to packmaker! This guide will help you begin using it to make Modded Minecraft modpacks.

Installing

You will need Python. Packmaker works on Python 3.4 and later.

If you have pip, just say `pip3 install packmaker` (or `pip3 install --user packmaker` if you run into permissions problems).

To install without pip, download packmaker from its PyPI page and run `python setup.py install` in the directory therein.

The best way to upgrade packmaker to a new version is by running `pip3 install -U packmaker`.

Configuring

Packmaker has a config file that you will need to provide. Packmaker will look in the following locations on the filesystem for this config file, in this order. The first file found is the one it uses:

- `./packmaker.conf`, in the current working directory.
- `${HOME}/packmaker.conf`, in your home directory.
- `${HOME}/.config/packmaker.conf`, the `.config` subdirectory of your home directory.

- `/usr/local/etc/packmaker.conf`
- `/etc/packmaker.conf`

Currently, this configuration file only contains some required parameters for dealing with the Curseforge API and some optional locations for where to store downloaded and built artifacts.

```
[curseforge]
authentication_token = <twitch authentication token>
moddb_filename = curseforge.db

[locations]
build = ./build
cache = ./build/cache
release = ./build/release
```

Packmaker needs to call the Curseforge API to retrieve mod information, and this api requires authentication. A twitch.tv account is required and your authentication oauth token can be generated using the Twitch Chat OAuth Password Generator. Copy the generated token into your config file.

Packmaker can download the enitre list of minecraft mods and generate a local database of information to be used later when building modpacks. The file this database is contained in also needs to be specified.

The *[locations]* section is optional but allows you to specify the locations where the files packmaker downloads and the artifacts it generates will be kept on your filesystem.

The *build* location is where packmaker puts its generated artifacts when building a pack. If not defined, the default build location is a *build* subdirectory of the current working directory.

The *cache* location is where packmaker places the files its downloads during the build process. Minecraft files, forge libraries, mods, etc are all downloaded and kept in this cache location. Packmaker tries to optimize its downloading and checks for files in this cache location before attempting to download a file so that it does not have to download the same file twice. If not defined, the default cache location is a *cache* subdirectory in the build location.

Finally, the *release* location is where packmaker will put the final artifacts generated by the build process. For example, the zip file you will want to upload to curseforge is placed here. If not defined, the default release location is a *release* subdirectory in the build location.

CurseforgeDB

With a configuration file in place, you can now generate a local database of information on minecraft mods. Run the following command to generate it:

```
packmaker updatedb
```

This command will use the curseforge api to retrieve the list of all available mods. As there are literally thousands of available mods, this command may take a few minutes to complete.

The Example Modpack

Packmaker modpacks are defined in a YAML file. By default, packmaker will look *packmaker.yml* in the current directory, although you can use any file name you like when you specify it on the packmaker command line.

Here is a short, quick example packmaker definition file:

```
---
name:  examplepack
title: Example Pack
version: 1.0.0

authors:
  - mcrewson

minecraft: 1.12.2
forge: 14.23.5.2838

mods:
  - jei:
  - journeymap:
  - mantle:
  - tinkers-construct:
  - the-one-probe:

files:
  - location: src/
```

Fields should be fairly obvious. The *Packmaker Definition Files* explains all of these options (and more) in details. But in summary, this example modpack is defined as follows:

- The modpack filename(s) will be a combination of the *name* and *version* field. For example, *examplepack-1.0.0.zip*
- The modpack metadata will include *title* and *authors* fields.
- This pack is for version 1.12.2 of Minecraft and version 14.23.5.2838 of Forge
- Packmaker will include the following mods in the pack: - Jei, Just Enough Items - Journeymap - Mantle - Tinker's Construct - The One Probe
- Packmaker will include all of the content of the *src* subdirectory in the modpack

With the modpack defined in the yaml file, next it need to be locked, and then built.

Locking a pack, with the *packmaker lock* command, will resolve all of the information in the yaml file into actionable items. Primarily, mods will be resolved to be downloadable urls. A *packmaker.lock* file will be generated with all of the resolved information in it, and subsequent build steps will reference this lock file.

After the pack is locked, you can build it into an installable modpack, suitable for running in your launcher, or uploading to the Curseforge website. Packmaker supports and is capable of building the pack into a number of different formats.

To build a curseforge compatible pack, run *packmaker build-curseforge*. You can upload the generate zip file (*build/release/examplepack-1.0.0.zip*) to curseforge, or import it into the Twitch.tv client or MultiMC client directly.

To build a server, suitable for multiplayer play, run *packmaker build-server*. You'll find the server files in *build/server/*. There is even a simple *start.sh* script you can use to run the server.

A local single player version of the pack can also be generated with *packmaker build-local*, found in *build/local/*. For pack testing purposes, there is even a convienent way to launch the pack in offline mode, *packmaker launch*.

1.2 Reference

This section contains reference materials for various parts of packmaker. To get started with packmaker as a new user, though, you may want to read the *Getting Started* guide first.

1.2.1 Packmaker Definition Files

Minecraft modpacks built using Packmaker are defined in a YAML file, typically called `packmaker.yml`[1].

The `packmaker.yml` file defines all of the metadata about a modpack, title, version, authors, etc, and all of the details about what is needed to be installed in a Minecraft instance to create the modpack. This include things like what version of Minecraft, what version of Forge, and any mods and/or resourcepacks to add to the instance.

There is an simple example of a `packmaker.yml` for a simple modpack in the packmaker git repository.

Introduction

Packmaker definition files are written in YAML syntax.

YAML is used because it is easier for humans to read and write than other common data formats like XML or JSON.

The definition file can be broken down into a number of different sections:

metadata All the pack metadata, including name, version, author, etc. Most of the metadata is required to properly define a modpack.

mods The list of mods to be installed in the modpack. Mods are optional, and if this section is omitted, no mods will be installed. (But what would a modpack be without mods?)

resourcepacks The list of resourcepacks to be installed in the modpack. Like mods, resourcepacks are optional, and if not are needed, this section can be omitted.

files The list of additional files to be installed in the modpack. Again, files are optional and can this section can be omitted if there are no files to install.

The following example packmaker definition file defines every single element, and every parameter for mods, resourcepacks, and files at least once.

```
---
name: examplepack
title: Example ModPack
version: 1.0.0
author:
  - mcrewson
  - crouth

minecraft: 1.12.2
forge: 14.23.5.2854

mods:
  - jei:
      version: latest
  - journeymap
  - mantle
  - the-one-probe
  - tinkers-construct:
      version: TConstruct-1.12.2-2.13.0.183.jar
      clientonly: false
      serveronly: false
      optional: false
      recommendation: starred
      selected: false
```

(continues on next page)

[1] `packmaker.yml` is the default name used for the this yaml file, and this reference document assumes the default filename is used, but you can use any filename you like, as long as you provide it on the command line.

(continued from previous page)

```
resourcepacks:
  - faithful-x32:
      version: Faithful 1.15.2-r1
      optional: false
      recommendation: starred
      selected: false

files:
  - location: src
    clientonly: false
    serveronly: false

routhio:
  html:
    - location: html
  launch:
    flags:
      - "-Dfml.ignoreInvalidMinecraftCertificates=true -XX:+UseG1GC -XX:UseSSE=3"
...
```

Metadata

Modpack metadata is needed to properly identify the modpack

name The name of the modpack. Currently used as part of the final filename for the released modpack file. Typically this should be short and simple, with no spaces or punctuation in the name, for better filenames.

Required.

title The title of the modpack. This should be the full title of the pack. For example, the title is what is used in a curseforge manifest file, so when you import your into twitch or multimc, this is the default name it will appear as.

Required.

version The version of the modpack. Can be any string, but would recommend your follow a consistent versioning pattern (semantic versioning is a good example). Incrementing the version of a pack is how launchers are able to detect and upgrade a modpack.

Version is also used in building the final filename for the released modpack file.

Required.

authors A list of names your want to credit with the creaton of the modpack. If only one name is needed, you can specify this as a string, rather than a list.

Some modpack formats, the curseforge format in particular, do not support more than one author in their metadata. When building modpacks in those formats, packmaker only uses the first name in the list.

Required.

minecraft This is the version of minecraft this pack is being built for. Any valid version string that Mojang has defined for Minecraft should be valid here, although it is recommended you use some of the more common versions:

- 1.7.10

- 1.10.2
- 1.11.2
- 1.12.2
- 1.13.2
- 1.14.4
- 1.15.2

This field is required.

forge The version of forge that the pack is being built for. Any valid version string that Forge has defined should be valid here, although it is recommended you use the latest for recommended version as defined by Forge.

This is an optoinal field. If not defined, your modpack will not include Forge, most likely resulting in a vanilla minecraft pack, one with no mods loaded in it.

Mod Defintions

`mods` is a list of minecraft mod definitions. Each definition requires the *slug name* of the mod, as found on the curseforge.com site.

You can find the *slug name* of a mod using packmaker's search function. Search results will contain both the full name of a mod and the *slug name*.

The *slug name* of a mod on curseforge can also be found in the mod's address on the site. For example, the curseforge page for the Tinker's Construct mod is:

https://www.curseforge.com/minecraft/mc-mods/tinkers-construct

The last part of the address, "**tinkers-construct**", is the *slug name* for this mod.

Each mod listed in this `packmaker.yml` file can be specified as either a string value, if no additional parameters are required for the mod, or as a dictionary, where you can supply additional parameters to control the specific version of the mod and other aspects of it within the modpack.

All of these parameters are optional.

version The specific version of the mod to be installed into the modpack. If not defined, packmaker will default to using the latest version of the mod available for the version of minecraft that the pack is built for.

The `version` parameter must contain the entire filename of the version of the mod desired, usually as specified on the curseforge site for the mod.

The value `latest` can also be used to explicitly specify that the latest version be installed.

clientonly A boolean value, indicating whether this mod is specific to the minecraft client. If true, packmaker will not install this mod into any server builds.

Optional, and defaults to false if not specified, meaning the mod will be installed in both client and server builds.

serveronly A boolean value, indicating whether this mod is specific to the minecraft server. If true, packmaker will only install this mod in server builds.

Optional, and default to false if not specified, meaning the mod will be installed in both client and server builds.

optional An optional parameter, only used for routhio builds. This will mark the mod as optional, and the routhio launcher will present it as an optional mod that can be installed but is not required to launch the modpack.

When building servers and other types of modpacks, for example curseforge modpacks, packmaker will ignore this parameter.

recommendation An optional parameter, only used for routhio builds. This will mark an optional mod as recommended, marking the mod in the launcher as a recommended optional mod or an optional mod to avoid. Should have the value *starred* or *avoid*. Any other value will not make sense to the routhio launcher and will cause the mod to be ignored. Not including this parameter on an optional mod will make the launcher neither recommend to install nor avoid it.

When building servers and other types of modpacks, for example curseforge modpacks, packmaker will ignore this parameter.

selected An optional parameter, only used for routhio builds. This will mark an optional mod as selected, making the optonal mod installed by default, unless the user explicitly chooses not to. Should have the value a boolean value, true or false.

When building servers and other types of modpacks, for example curseforge modpacks, packmaker will ignore this parameter.

Resourcepack Definitions

Resourcepacks are always minecraft client only elements. They will be ignored when building servers for the pack. So there is no need for `clientonly` or `serveronly` parameters for a resourcepack.

version The specific version of the resourcepack to be installed into the modpack. If not defined, packmaker will default to using the latest version of the resoourcepack available for the version of minecraft that the pack is built for.

The `version` parameter must contain the entire filename of the version of the resourcepack desired, usually as specified on the curseforge site for the resourcepack.

The value `latest` can also be used to explicitly specify that the latest version be installed.

optional An optional parameter, only used for routhio builds. This will mark the resourcepack as optional, and the routhio launcher will present it as an optional resourcepack that can be installed but is not required to launch the modpack.

When building other types of modpacks, for example curseforge modpacks, packmaker will ignore this parameter.

recommendation An optional parameter, only used for routhio builds. This will mark an optional resourcepack as recommended, marking the resourcepack in the launcher as a recommended optional resourcepack or an optional resourcepack to avoid. Should have the value *starred* or *avoid*. Any other value will not make sense to the routhio launcher and will cause the resourcepack to be ignored. Not including this parameter on an optional resourcepack will make the launcher neither recommend to install nor avoid it.

When building other types of modpacks, for example curseforge modpacks, packmaker will ignore this parameter.

selected An optional parameter, only used for routhio builds. This will mark an optional resourcepack as selected, making the optonal resourcepack installed by default, unless the user explicitly chooses not to. Should have the value a boolean value, true or false.

When building other types of modpacks, for example curseforge modpacks, packmaker will ignore this parameter.

Files

The files section of the `packmaker.yml` file specifies the local files to be included into the modpack. Typically these would be an additional configurations, scripts, resources, etc that you want to add to the minecraft install when it is installed.

This section is a list of file locations, typically subfolders or directories within your packmaker project. Each location in the list should be unique and not overlap another location, as all locations will be combined in the final poduct.

The contents of each location folder will be copied into the minecraft folder of the built instance, with no alterations or attempt to preserve what is already stored in the destination, so be care not to accidently overwrite any files that may have already been installed by minecraft itself or mod installations.

A typical use for multiple file locations is to specify some files be installed only on client builds, and other only installed on server builds. For example:

```
files:
  - location: src-all
  - location: src-client
    clientonly: true
  - location: src-server
    serveronly: true
```

Each location in the files list can have the following parameters.

location The location, relative to the `packmaker.yml` file itself, of the local files to be included in the modpack.

If this location does not physically exist when building the modpack, a warning is printing, by this location will be ignored.

clientonly An optional boolean flag (true or false), indicating that this location of files should only be included with client builds.

Defaults to false, meaning the location will be included in both client and server build, unless the corresponding `serveronly` parameter is specified.

serveronly An optional boolean flag (true or false), indicating that this location of files should only be included with server builds.

Defaults to false, meaning the location will be included in both client and server build, unless the corresponding `clientonly` parameter is specified.

Routhio

The routhio section of the `packmaker.yml` file is specific to only one type of modpack build, a modpack for the routhio launcher. All other build types will ignore this section.

This section includes the following elements:

html A location of additional html files, typically a local folder with the project, that will be displayed within the launcher when a user selected this specific modpack.

launch Additional options the launcher will use when launching a minecraft instance. Currently, the only launcher option that can be specified here is `flags`, which is used to provide additional java command line arguments on the minecraft process.

1.2.2 Command-Line Interface

Commands

build-curseforge

```
packmaker build-curseforge [-h] [--build-dir BUILD_DIR]
                                [--release-dir RELEASE_DIR]
                                [--cache-dir CACHE_DIR]
                                [--release-format {zip,tgz}]
                                [lockfile [lockfile ...]]

Build a curseforge compatible modpack.

positional arguments:
  lockfile              modpack lock file

optional arguments:
  -h, --help            show this help message and exit
  --build-dir BUILD_DIR, -b BUILD_DIR
                        base directory for build artifacts
  --release-dir RELEASE_DIR, -r RELEASE_DIR
                        base directory for release artifacts
  --cache-dir CACHE_DIR
                        base directory for cached artifacts
  --release-format {zip,tgz}
                        archive format for release package
```

Describe cmd here

Optional command flags:

build-local

```
packmaker build-local [-h] [--build-dir BUILD_DIR]
                           [--release-dir RELEASE_DIR]
                           [--cache-dir CACHE_DIR]
                           [--release-format {zip,tgz}]
                           [lockfile [lockfile ...]]

Build a local installation

positional arguments:
  lockfile              modpack lock file

optional arguments:
  -h, --help            show this help message and exit
  --build-dir BUILD_DIR, -b BUILD_DIR
                        base directory for build artifacts
  --release-dir RELEASE_DIR, -r RELEASE_DIR
                        base directory for release artifacts
  --cache-dir CACHE_DIR
                        base directory for cached artifacts
  --release-format {zip,tgz}
                        archive format for release package
```

Describe cmd here

Optional command flags:

build-routhio

```
packmaker build-routhio [-h] [--build-dir BUILD_DIR]
                            [--release-dir RELEASE_DIR]
                            [--cache-dir CACHE_DIR]
                            [--release-format {zip,tgz}]
                            [lockfile [lockfile ...]]

Build a Routh.io compatible modpack.

positional arguments:
  lockfile              modpack lock file

optional arguments:
  -h, --help            show this help message and exit
  --build-dir BUILD_DIR, -b BUILD_DIR
                        base directory for build artifacts
  --release-dir RELEASE_DIR, -r RELEASE_DIR
                        base directory for release artifacts
  --cache-dir CACHE_DIR
                        base directory for cached artifacts
  --release-format {zip,tgz}
                        archive format for release package
```

Describe cmd here

Optional command flags:

build-server

```
packmaker build-server [-h] [--build-dir BUILD_DIR]
                           [--release-dir RELEASE_DIR]
                           [--cache-dir CACHE_DIR]
                           [--release-format {zip,tgz}]
                           [lockfile [lockfile ...]]

Build a server modpack.

positional arguments:
  lockfile              modpack lock file

optional arguments:
  -h, --help            show this help message and exit
  --build-dir BUILD_DIR, -b BUILD_DIR
                        base directory for build artifacts
  --release-dir RELEASE_DIR, -r RELEASE_DIR
                        base directory for release artifacts
  --cache-dir CACHE_DIR
                        base directory for cached artifacts
  --release-format {zip,tgz}
                        archive format for release package
```

Describe cmd here

Optional command flags:

convert

```
packmaker convert [-h] manifest [packdef]

Convert a curse/twitch modpack manifest json file to a packmaker yaml packdef file

positional arguments:
  manifest    manifest json file
  packdef     output packdef file

optional arguments:
  -h, --help  show this help message and exit
```

Describe cmd here

Optional command flags:

findupdates

```
packmaker findupdates [-h] [--build-dir BUILD_DIR]
                           [--release-dir RELEASE_DIR]
                           [--cache-dir CACHE_DIR]
                           [--release-format {zip,tgz}]
                           [-f {csv,json,table,value,yaml}] [-c COLUMNS]
                           [--max-width <integer>] [--fit-width]
                           [--print-empty]
                           [--quote {all,minimal,none,nonnumeric}]
                           [--noindent] [--sort-column SORT_COLUMN]
                           [lockfile [lockfile ...]]

Search curseforge for newer/updated versions of addons.

positional arguments:
  lockfile              modpack lock file

optional arguments:
  -h, --help            show this help message and exit
  --build-dir BUILD_DIR, -b BUILD_DIR
                        base directory for build artifacts
  --release-dir RELEASE_DIR, -r RELEASE_DIR
                        base directory for release artifacts
  --cache-dir CACHE_DIR
                        base directory for cached artifacts
  --release-format {zip,tgz}
                        archive format for release package

output formatter:
  output formatter options

  -f {csv,json,table,value,yaml}, --format {csv,json,table,value,yaml}
                        the output format, defaults to table
```

(continues on next page)

(continued from previous page)

```
  -c COLUMNS, --column COLUMNS
                        specify the column(s) to include, can be repeated
  --sort-column SORT_COLUMN
                        specify the column(s) to sort the data (columns
                        specified first have a priority, non-existing columns
                        are ignored), can be repeated

Table formatter:
  --max-width <integer>
                        maximum display width, <1 to disable. You cal also use
                        the MAX_DISPLAY_WIDTH environment variable, but the
                        parameter takes precedence.
  --fit-width           Fit the table to the display width. Implied if --max-
                        width greater than 0. Set the environment variable
                        FIT_WIDTH=1 to always enable
  --print-empty         Print empty table iof there is no data to show.

CSV formatter:
  --quote {all,minimal,none,nonnumeric}
                        when to include quotes, default to nonnumeric

json formatter:
  --noindent            whether to disable indenting the JSON
```

Describe cmd here

Optional command flags:

info

```
packmaker info [-h] [--build-dir BUILD_DIR] [--release-dir RELEASE_DIR]
                       [--cache-dir CACHE_DIR] [--release-format {zip,tgz}]
                       [-f {json,shell,table,value,yaml}] [-c COLUMNS]
                       [--max-width <integer>] [--fit-width] [--print-empty]
                       [--noindent] [--prefix PREFIX]
                       [lockfile [lockfile ...]]

Display information about the modpack.

positional arguments:
  lockfile              modpack lock file

optional arguments:
  -h, --help            show this help message and exit
  --build-dir BUILD_DIR, -b BUILD_DIR
                        base directory for build artifacts
  --release-dir RELEASE_DIR, -r RELEASE_DIR
                        base directory for release artifacts
  --cache-dir CACHE_DIR
                        base directory for cached artifacts
  --release-format {zip,tgz}
                        archive format for release package

output formatter:
  output formatter options
```

(continues on next page)

(continued from previous page)

```
  -f {json,shell,table,value,yaml}, --format {json,shell,table,value,yaml}
                        the output format, defaults to table
  -c COLUMNS, --column COLUMNS
                        specify the column(s) to include, can be repeated

Table formatter:
  --max-width <integer>
                        maximum display width, <1 to disable. You cal also use
                        the MAX_DISPLAY_WIDTH environment variable, but the
                        parameter takes precedence.
  --fit-width           Fit the table to the display width. Implied if --max-
                        width greater than 0. Set the environment variable
                        FIT_WIDTH=1 to always enable
  --print-empty         Print empty table iof there is no data to show.

json formatter:
  --noindent            whether to disable indenting the JSON

shell formatter:
  a format a UNIX shell can parse (variable="value")

  --prefix PREFIX       add a prefix to all variable names
```

Describe cmd here

Optional command flags:

launch

```
packmaker launch [-h] [--build-dir BUILD_DIR]
                        [--release-dir RELEASE_DIR] [--cache-dir CACHE_DIR]
                        [--release-format {zip,tgz}]
                        [lockfile [lockfile ...]]

Launch a local installation

positional arguments:
  lockfile              modpack lock file

optional arguments:
  -h, --help            show this help message and exit
  --build-dir BUILD_DIR, -b BUILD_DIR
                        base directory for build artifacts
  --release-dir RELEASE_DIR, -r RELEASE_DIR
                        base directory for release artifacts
  --cache-dir CACHE_DIR
                        base directory for cached artifacts
  --release-format {zip,tgz}
                        archive format for release package
```

Describe cmd here

Optional command flags:

lock

```
packmaker lock [-h] [packdef [packdef ...]]

Lock the modpack. Find mod download urls, generate a packmaker.lock file.

positional arguments:
  packdef     modpack definition file

optional arguments:
  -h, --help  show this help message and exit
```

Describe cmd here

Optional command flags:

modsinfo

```
packmaker modsinfo [-h] [--build-dir BUILD_DIR]
                        [--release-dir RELEASE_DIR] [--cache-dir CACHE_DIR]
                        [--release-format {zip,tgz}]
                        [-f {csv,json,table,value,yaml}] [-c COLUMNS]
                        [--max-width <integer>] [--fit-width]
                        [--print-empty]
                        [--quote {all,minimal,none,nonnumeric}] [--noindent]
                        [--sort-column SORT_COLUMN]
                        [lockfile [lockfile ...]]

Display information about the mods in the modpack.

positional arguments:
  lockfile              modpack lock file

optional arguments:
  -h, --help            show this help message and exit
  --build-dir BUILD_DIR, -b BUILD_DIR
                        base directory for build artifacts
  --release-dir RELEASE_DIR, -r RELEASE_DIR
                        base directory for release artifacts
  --cache-dir CACHE_DIR
                        base directory for cached artifacts
  --release-format {zip,tgz}
                        archive format for release package

output formatter:
  output formatter options

  -f {csv,json,table,value,yaml}, --format {csv,json,table,value,yaml}
                        the output format, defaults to table
  -c COLUMNS, --column COLUMNS
                        specify the column(s) to include, can be repeated
  --sort-column SORT_COLUMN
                        specify the column(s) to sort the data (columns
                        specified first have a priority, non-existing columns
                        are ignored), can be repeated
```

(continues on next page)

(continued from previous page)

```
Table formatter:
  --max-width <integer>
                        maximum display width, <1 to disable. You cal also use
                        the MAX_DISPLAY_WIDTH environment variable, but the
                        parameter takes precedence.
  --fit-width           Fit the table to the display width. Implied if --max-
                        width greater than 0. Set the environment variable
                        FIT_WIDTH=1 to always enable
  --print-empty         Print empty table iof there is no data to show.

CSV formatter:
  --quote {all,minimal,none,nonnumeric}
                        when to include quotes, default to nonnumeric

json formatter:
  --noindent            whether to disable indenting the JSON
```

resourcepacksinfo

```
packmaker resourcepacksinfo [-h] [--build-dir BUILD_DIR]
                                 [--release-dir RELEASE_DIR]
                                 [--cache-dir CACHE_DIR]
                                 [--release-format {zip,tgz}]
                                 [-f {csv,json,table,value,yaml}]
                                 [-c COLUMNS]
                                 [--max-width <integer>]
                                 [--fit-width]
                                 [--print-empty]
                                 [--quote {all,minimal,none,nonnumeric}]
                                 [--noindent]
                                 [--sort-column SORT_COLUMN]
                                 [lockfile [lockfile ...]]

Display information about the resourcepacks in the modpack.

positional arguments:
  lockfile              modpack lock file

optional arguments:
  -h, --help            show this help message and exit
  --build-dir BUILD_DIR, -b BUILD_DIR
                        base directory for build artifacts
  --release-dir RELEASE_DIR, -r RELEASE_DIR
                        base directory for release artifacts
  --cache-dir CACHE_DIR
                        base directory for cached artifacts
  --release-format {zip,tgz}
                        archive format for release package

output formatter:
  output formatter options

  -f {csv,json,table,value,yaml}, --format {csv,json,table,value,yaml}
                        the output format, defaults to table
```

(continues on next page)

(continued from previous page)

```
  -c COLUMNS, --column COLUMNS
                        specify the column(s) to include, can be repeated
  --sort-column SORT_COLUMN
                        specify the column(s) to sort the data (columns specified␣
↪first have a priority, non-existing columns are ignored), can be repeated

Table formatter:
  --max-width <integer>
                        maximum display width, <1 to disable. You cal also use the␣
↪MAX_DISPLAY_WIDTH environment variable, but the parameter takes precedence.
  --fit-width           Fit the table to the display width. Implied if --max-width␣
↪greater than 0. Set the environment variable FIT_WIDTH=1 to always enable
  --print-empty         Print empty table iof there is no data to show.

CSV formatter:
  --quote {all,minimal,none,nonnumeric}
                        when to include quotes, default to nonnumeric

json formatter:
  --noindent            whether to disable indenting the JSON
```

Describe cmd here

Optional command flags:

search

```
packmaker search [-h] [-f {csv,json,table,value,yaml}] [-c COLUMNS]
                        [--max-width <integer>] [--fit-width] [--print-empty]
                        [--quote {all,minimal,none,nonnumeric}] [--noindent]
                        [--sort-column SORT_COLUMN]
                        searchstring [searchstring ...]

Search curseforge for mods.

positional arguments:
  searchstring          search string

optional arguments:
  -h, --help            show this help message and exit

output formatter:
  output formatter options

  -f {csv,json,table,value,yaml}, --format {csv,json,table,value,yaml}
                        the output format, defaults to table
  -c COLUMNS, --column COLUMNS
                        specify the column(s) to include, can be repeated
  --sort-column SORT_COLUMN
                        specify the column(s) to sort the data (columns
                        specified first have a priority, non-existing columns
                        are ignored), can be repeated

Table formatter:
  --max-width <integer>
```

(continues on next page)

(continued from previous page)

```
                        maximum display width, <1 to disable. You cal also use
                        the MAX_DISPLAY_WIDTH environment variable, but the
                        parameter takes precedence.
  --fit-width           Fit the table to the display width. Implied if --max-
                        width greater than 0. Set the environment variable
                        FIT_WIDTH=1 to always enable
  --print-empty         Print empty table iof there is no data to show.

CSV formatter:
  --quote {all,minimal,none,nonnumeric}
                        when to include quotes, default to nonnumeric

json formatter:
  --noindent            whether to disable indenting the JSON
```

Describe cmd here

Optional command flags:

updatedb

```
packmaker updatedb [-h] [--ignore-mods] [--ignore-resourcepacks]

Download and compile a new mods database from curseforge.

optional arguments:
  -h, --help            show this help message and exit
  --ignore-mods         Do not scan for mods when updating the curseforge db
  --ignore-resourcepacks
                        Do not scan for resourcepacks when updating the curseforge db
```

Describe cmd here

Optional command flags:

help

```
packmaker help [-h] cmd

Print detailed help for another command

positional arguments:
  cmd         name of the command

optional arguments:
  -h, --help  show this help message and exit
```

Describe cmd here

Optional command flags:

Global Flags

Packmaker has a few "global" flags that affect all commands. These must appear between the executable name (`packmaker`) and the command—for example, `packmaker -v search ...`.

1.2.3 Configuration

tbd...

1.3 FAQ

Here are some answers to frequently-asked questions from IRC and elsewhere. Got a question that isn't answered here? Try the discord server, or filing an issue in the bug tracker.

1.3.1 How do I...

tbd...

1.3.2 Why does packmaker...

tbd...