# Racket in Gentoo

Maciej "XGQT" Barć

# Outline

# About the author

Maciej "XGQT" Barć <xgqt@gentoo.org>

wiki.gentoo.org/wiki/User:Xgqt

PGP: `9B0A 4C5D 02A3 B43C 9D6F D6B1 14D7 4A1F 43A6 AC3C`

- Racket user since 2021
- Gentoo Linux user since 2017
- Gentoo developer (since 2020),
  member of: dotnet, emacs, math, ml, nim, scheme, sci

# Agenda

- First about Racket tools for Gentoo
- then about tools for Racket.

# Gentoo

- source based distribution
- fine-grained control over the operating system
  - profiles (eg `desktop/plasma`)
  - USE flags
  - program alternatives
- proper system "base"?
  Daphile, Porteus Kiosk, ChromeOS, LigurOS, Calculate, ~~Sabayon (R.I.P.)~~

# Portage

- uses USE flags to customize the features and dependencies of packages
- supports package masking, unmasking, and blocking to manage conflicts and ensure compatibility between packages
- uses ebuilds, which are scripts that describe how to download, configure, compile, and install software packages
- ebuilds are stored in a centralized repository, and Portage can automatically download and update packages from this repository

# PMS

- set of guidelines that describe how ebuilds should be written and how they should interact with the package manager
- defines a set of standard variables and functions that ebuilds should use to interact with operating system
- describes correct package versioning, dependencies, and metadata, which helps ensure that packages are correctly installed and configured on Gentoo systems
- is periodically updated to reflect changes in the Gentoo ecosystem and to address issues that arise during development and maintenance of packages

# PMS / Phases

- `pkg`
  - `pretend`
  - `setup`
- `src` (skipped for binary packages)
  - `unpack`
  - `prepare`
  - `configure`
  - `compile`
  - `test`
    (optional, only when tests are enabled - `FEATURES=test`)
  - `install`
- `pkg`
  - `preinst`
  - `postinst`

# Ebuilds / racket.eclass / raco install

```
# @FUNCTION: raco_install
# @USAGE: [arg] ...
# @DESCRIPTION:
# Calls "raco pkg install" with given options.
raco_install() {
    debug-print-function ${FUNCNAME} "${@}"

    local raco_opts=(
        --batch
        --deps force
        --force
        --jobs "$(makeopts_jobs)"
        --no-cache
    )
    eraco pkg install "${raco_opts[@]}" "${@}"
}
```

# Ebuilds / racket.eclass / racket src compile

```
# @FUNCTION: racket_src_compile
# @DESCRIPTION:
# Default src_compile:
# Executes "raco_temporary_install" and conditionally
# "scribble_system_docs".
racket_src_compile() {
    raco_temporary_install

    if [[ ${_do_scrbl} -eq 1 ]] && use doc ; then
        scribble_system_docs
    fi
}
```

# Ebuilds / example / ebuild

```
EAPI=8
GH_DOM="github.com"
GH_REPO="lexi-lambda/threading"
GH_COMMIT="1f3b630d3ed78edc6d883f9d99305158d6b67623"

inherit gh racket

DESCRIPTION="The threading Racket package"
HOMEPAGE="https://github.com/lexi-lambda/threading"
S="${S}/threading"

LICENSE="ISC"
SLOT="0"
KEYWORDS="~amd64 ~arm ~ppc ~ppc64 ~x86"

RDEPEND="dev-racket/threading-doc
    dev-racket/threading-lib"
BDEPEND="${RDEPEND}"
```

# Ebuilds / example / cache

```
BDEPEND=dev-racket/threading-doc dev-racket/threading-lib
>=dev-scheme/racket-8.1:=[-minimal]
sys-apps/racket-where
DEFINED_PHASES=compile install postinst prepare prerm test
DESCRIPTION=The threading Racket package
EAPI=8
HOMEPAGE=https://github.com/lexi-lambda/threading
INHERIT=gh racket
IUSE=doc
KEYWORDS=~amd64 ~arm ~ppc ~ppc64 ~x86
LICENSE=ISC
RDEPEND=dev-racket/threading-doc dev-racket/threading-lib
>=dev-scheme/racket-8.1:=[-minimal]
SLOT=0
_eclasses_=gh 2efd973430985d5fee2990854013b2ee
racket-common 8d9a0235222199d9692a3941181901c2
racket  97da5091f435674bf3acc717a35ee8a9
_md5_=01b7f6d6b9680ca42b5c5cb23d626989
```

# The Gentoo-Racket project

Gentoo-Racket is a unofficial project that aims to bring first-class support for Racket programming language ecosystem to Gentoo.

gentoo-racket.gitlab.io

# Notable projects

- Gentoo-Racket-Overlay —— Gentoo overlay with Racket packages (standalone repository)
- collector2 —— generate Gentoo ebuilds from the Racket Catalog
- ebuild —— library to ease automatic ebuild creation
- vdb —— Racket interface to the Portage VDB
- pmsf —— Package Manager Specification formatting and parsing
- eclass2scrbl —— convert Gentoo Eclasses to Scribble documents
- racket-where —— small tool to find install locations of Racket packages

# Collector2

- started in 2021 (before I was officially in Gentoo)
- simple script evolved into a tool
- later the Racket-Ebuild library was extracted from it

- generates ebuilds from a given Racket Package Catalog
- can filter (un)wanted packages
- applies several modifications to dependencies (remove base & core packages)

# Racket-Ebuild

- creating ebuild scripts
- initially a part of Collector2, later split off into it's own package
- OOP-style
- uses some outdated structures that should be replaced by PMSF's ones

# Racket-Ebuild / Example

```
(define test-ebuild
  (new ebuild%
       [inherits '("git-r3")]))
(define test-metadata
  (new metadata%
       [maintainers
        (list
         (maintainer 'project #f "null@gentoo.org" #f #f))]))
(define test-pkg
  (new package%
       [CATEGORY "sys-devel"]
       [PN       "test-pkg"]
       [ebuilds  (hash (simple-version "0") test-ebuild)]
       [metadata test-metadata]))
(define test-repository
  (new repository%
       [name     "test-repo"]
       [packages (list test-pkg)]))
```

# PMSF

- library to support different cases of special syntax used by ebuilds
- based on the Package Manager Specification (PMS)

# PMSF / Example / SRC URI

```
(string->psrc-uri
 "
      mirror://openssl/source/openssl-1.1.1q.tar.gz
      verify-sig? (
          mirror://openssl/source/openssl-1.1.1q.tar.gz.asc
      )
     ")
```

```
(psrc-uri
 (list (psource
        (url "mirror" ... #f) #f)
       (pcondition "verify-sig"
                   (list
                    (psource (url "mirror" ... #f) #f)))))
```

# PMSF / Example / DEPEND

```
(string->pdepend "ssl? ( dev-libs/openssl:= )")
```

```
(pdepend
 (list
  (pcondition
   "ssl"
   (list
    (pdependency #f
                 #f
                 (pcomplete "dev-libs" (pfull "openssl" #f))
                 #f
                 #f
                 "="
                 '()))))))
```

# VDB

- Racket interface to the Portage "VDB" (`/var/db/pkg/`), which is a place where Portage stores information about installed packages
- uses Racket-PMSF to parse Portage metadata
- mostly written in Typed-Racket
  (currently without the "write" functions)

# VDB / Example / Dependencies

```
(vdb-package-info-ref "sys-process/htop-3.2.2" 'RDEPEND)
```

```
(pdepend
 (list
  (pdependency
   #f
   #f
   (pcomplete "sys-libs" (pfull "ncurses" #f))
   #f
   (pslot "0" "6")
   "="
   (list (pdepflag "unicode" #f 'enabled)))
  ...))
```

# VDB / Example / Files

```
(vdb-package-info-ref "sys-process/htop-3.2.2" 'CONTENTS)
```

```
(hash #<path:/usr/share/doc/htop-3.2.2/README.bz2>
      (content 'file
               #<path:/usr/share/doc/htop-3.2.2/README.bz2>
               "ddba4d9b017593723d2ec9e38e1e463e"
               1678524865
               #f)
      ...)
```

# Beyond Gentoo?

- new software not strictly conforming to PMS can be created using libraries designed for Gentoo
- proof-of-concept binary package manager for MS Windows using powershell instead of bash

# Outside Gentoo, inside Racket

pkgd.racket-lang.org/xgqt@riseup.net

Smaller packages:

- dirname — `basename` and `dirname` functions
- typed-colorize — typed interface to `colorize`
- ini — Racket parser for Ini and UNIX Conf files
- upi — Racket library inspired by UNIX-like tools
- ziptie — miscellaneous "glue" (`distro`, `git`, `monorepo`, etc.)
- mike — micro `make` replacement (before `zuo` ;-P)

# Make-Makefile (Ziptie)

- attempt to utilize `make` for faster package development
- generate `Makefile` files based on packages contained within a repository
- superseded by Req
- future plans: rewrite to allow exporting to Zuo

# Req

Req was inspired by Python's `requirements.txt` files which list all of dependencies a particular Python project uses.

Req offers a convenient way to manage dependencies of large Racket packages.

- install only external dependencies,
- install all local packages at once,
- use a special catalogs repository for the project,
- save the extra packages You use for repository maintenance.

# Req / Showcase

## Commands

```
raco req --everything
```

or for short:

```
raco req -A
```

## Collector2 example commands

```
raco req -R
raco req -A
```

# BumpV

BumpV is a tool that helps to quickly make new version releases of Racket-based projects.

- finds a special `version.rkt` file
- increases the version number
- modifies the specified files with old version information (for example the `info.rkt` files)
- additionally can create a git commit and tags

# BumpV / Showcase

## Commands

```
bumpv -ctv 3 1 1
```

## PMSF example output

```
"bumpv", version 1.2.1, running on Racket 8.8
Version file: /home/.../pmsf-lib/pmsf/version.rkt
Old VERSION:  3.1.0
New VERSION:  3.1.1
Creating a git commit...
Creating a git tag...
```

# Improvements

- pkg server modernization
- extract versions and/or tags
- description is still being defined by the user
- tox / stack for Racket?
  (regression testing across Racket & package versions)

# Thanks

Special thanks to many Racket contributors

- Matthew Flatt
- Sam Tobin-Hochstadt
- Jens Axel Søgaard
- Sorawee Porncharoenwase
- Bogdan Popa
- Alexis King

and Gentoo developers making this possible with their hard work

- Ulrich Müller
- Michał Górny
- Arthur Zamarin
- Tim Harder
- Zac Medico