Package: ravemanager (via r-universe)

November 22, 2024

Title Manage 'RAVE' Packages

Version 1.0.44

Description Manages installation, upgrade, and removal of packages and corresponding dependence for the project 'RAVE' (R Analysis and Visualization for 'iEEG').

License MIT + file LICENSE

Language en-US

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

URL `https://dipterix.org/ravemanager/`,
`http://dipterix.org/ravemanager/`

Imports utils

Suggests learnr, pkgload, testthat (>= 3.0.0)

Config/testthat/edition 3

Repository https://rave-ieeg.r-universe.dev

RemoteUrl https://github.com/dipterix/ravemanager

RemoteRef HEAD

RemoteSha cbb99a5e526e71ef76e378ad635690ffc853e9ee

Contents

`configure-python` *Install & Configure 'python' environment for 'RAVE'*

Description

Installs 'python' environment for 'RAVE'

Usage

```
validate_python(verbose = TRUE)

configure_python(python_ver = "3.11", verbose = TRUE)

remove_conda(ask = TRUE)

ensure_rpymat()
```

Arguments

`verbose`	whether to verbose messages
`python_ver`	python version; default is automatically determined. If a specific python version is needed, please specify explicitly, for example, `python_ver='3.8'`
`ask`	whether to ask before resetting the 'python' environment

Details

Use `ravemanager::configure_python()` to install and configure python environment using `miniconda`. The conda binary and environment will be completely isolated. This means the installation will be safe and it will not affect any existing configurations on the computer.

In this isolated environment, the following packages will be installed: `numpy`, `scipy`, `pandas`, `h5py`, `jupyterlab`, `pynwb`, `mat73`, `mne`. You can always add more `conda` packages via `rpymat::add_packages(...)` or `pip` packages via `rpymat::add_packages(..., pip = TRUE)`.

To use the 'python' environment, please run `rpymat::ensure_rpymat()` to activate in your current session. If you are running via `'RStudio'`, open any 'python' script and use `'ctrl/cmd+enter'` to run line-by-line. To switch from `R` to `python` mode, using command `rpymat::repl_python()`

A `jupyterlab` will be automatically installed during the configuration. To launch the `jupyterlab`, use `rpymat::jupyter_launch()`

If you want to remove this `conda` environment, use R command `rpymat::remove_conda()`. This procedure is absolutely safe and will not affect your other installations.

Examples

```
## Not run:

# -------- Install & Configure python environment for RAVE --------
ravemanager::configure_python()

# Add conda packages
rpymat::add_packages("numpy")

# Add pip packages
rpymat::add_packages("nipy", pip = TRUE)

# -------- Activate RAVE-python environment --------
rpymat::ensure_rpymat()

# run script from a temporary file
f <- tempfile(fileext = ".py")
writeLines(c(
  "import numpy",
  "print(f'numpy installed: {numpy.__version__}')"
), f)
rpymat::run_script(f)

# run terminal command within the environment
rpymat::run_command("pip list")

# run python interactively, remember to use `exit` to exit
# python mode
rpymat::repl_python()




## End(Not run)
```

`debug_info` *Print 'RAVE' debugging information*

Description

This function will generate a report. Please include the report when you file an issue. Useful for reporting issues to 'RAVE' developers.

Usage

```
debug_info(max_lines)
```

Arguments

`max_lines`	maximum number of log entries to print

`export_logs` *Print out 'RAVE' session log*

Description

Print out 'RAVE' session log

Usage

```
export_logs(
  session = NULL,
  modules = NULL,
  max_lines = Sys.getenv("RAVEMANAGER_BUGREPORT_MAX", "200"),
  verbose = TRUE
)
```

Arguments

`session`	'RAVE' session string; default is the most recent (active) session
`modules`	which module to read; default is all
`max_lines`	maximum number of lines to read; default is 200
`verbose`	whether to print out the log; default is true

Value

characters of log

`find_packages_with_empty_files` *Find packages with empty files*

Description

Check whether packages are installed correctly. In rare cases (possibly network issues), packages downloaded contain empty files. This function provides a method to check empty files in packages.

Usage

```
find_packages_with_empty_files(lib = get_libpaths(check = TRUE))
```

Arguments

`lib`	library path where packages are installed; default is set to user library path.

Value

A list of packages (and files) containing empty files.

Examples

```
find_packages_with_empty_files()
```

`install_packages` *Install/Update R or Python packages to RAVE environment*

Description

Install/Update R or Python packages to RAVE environment

Usage

```
add_r_package(
  pkg,
  lib = get_libpaths(check = TRUE),
  repos = get_mirror(),
  type = getOption("pkgType"),
  ...,
  INSTALL_opts = "--no-lock"
)

add_py_package(pkg, method = c("pip", "conda"))
```

Arguments

`pkg`	name of the package
`repos`, `lib`, `type`, `INSTALL_opts`, `...`	internally used
`method`	whether to use `'pip'` or `'conda'`; default is `'pip'`

Value

Nothing

Examples

```
## Not run:


# ---- R -------------------------------------------------------
# Install R packages (CRAN, BioC, or RAVE's repository)
add_r_package("ravebuiltins")
add_r_package("rhdf5")

# Install from Github (github.com/dipterix/threeBrain)
add_r_package("dipterix/threeBrain")

# Install Github branch
add_r_package("dipterix/threeBrain@custom-electrode-geom")

# ---- Python --------------------------------------------------

# Normal pypi packages
add_py_package("threebrainpy")

# Add through conda
add_py_package("fftw", method = "conda")



## End(Not run)
```

RAVE-install *Install or upgrade 'RAVE'*

Description

Installs the newest version of 'RAVE' and its dependence packages; executes the scripts to finalize installation to update configuration files.

Usage

```
add_shortcuts()

finalize_installation(
  packages = NULL,
  upgrade = c("config-only", "ask", "always", "never", "data-only"),
  async = FALSE,
  ...
)

clear_cache()
```

```
install(
  allow_cache = FALSE,
  upgrade_manager = FALSE,
  finalize = TRUE,
  force = FALSE,
  python = FALSE,
  migrate_packages = FALSE,
  lib_path = NA,
  ...
)

update_rave(
  allow_cache = FALSE,
  upgrade_manager = FALSE,
  finalize = TRUE,
  force = FALSE,
  python = FALSE,
  migrate_packages = FALSE,
  lib_path = NA,
  ...
)

upgrade_installer(reload = TRUE)
```

Arguments

`packages`	packages to run finalizing installation scripts
`upgrade`	upgrade type
`async`	whether to execute finalizing installation scripts in other processes
`...`	passed to internal functions
`allow_cache`	whether to allow cache; default is true
`upgrade_manager`	whether to upgrade the installer (`ravemanager`) before updating other packages
`finalize`	whether to run finalizing installation scripts
`force`	whether to force updating packages even the installed have the latest versions
`python`	whether to install python; default is false
`migrate_packages`	whether to migrate (copy) packages installed in the system path to the user library; used as alternative option when there are existing packages installed in the system library path that could be hard to resolve or out-dated; default is `FALSE`
`lib_path`	library path where 'RAVE' should be installed; default is automatically determined.
`reload`	whether to reload `ravemanager` after installation; default is true. This tries to load the upgraded `ravemanager` without restarting the R session; however, this solution not always works. In such case, restarting R session is always the solution.

Value

Nothing

`ravemanager_version` *Get current 'RAVE' installer's version*

Description

Get current 'RAVE' installer's version

Usage

```
ravemanager_version()
```

Value

Returns the `ravemanager` version

`run_tutorials` *Run tutorials*

Description

Run tutorials

Usage

```
run_tutorials(topic = NULL, ...)
```

Arguments

`topic`	integers of which topic to launch, leave it blank, then 'RAVE' will ask you to select a topic
`...`	other parameters to pass to `shiny::runApp()`

Examples

```
## Not run:

ravemanager::run_tutorials()


## End(Not run)
```

`system_requirements` *Get system requirements for 'RAVE'*

Description

Get system requirements for 'RAVE'

Usage

```
system_requirements(
  os = NULL,
  os_release = NULL,
  curl = Sys.which("curl"),
  sudo = FALSE,
  ...
)
```

Arguments

`os, os_release`	operating system and release version, currently only supports `'ubuntu'`, `'centos'`, `'redhat'`, `'opensuse'`, and `'sle'`; see `https://github.com/rstudio/r-system-requirements#operating-systems`
`curl`	the location of the curl binary on your system
`sudo`	whether to pre-pend `'sudo'` to the commands
`...`	reserved for future use

Examples

```
if("remotes" %in% loadedNamespaces()) {
# Please check your operating system & version!!!

# =============== On Ubuntu Linux ===============

# Ubuntu 20
ravemanager::system_requirements("ubuntu", "20")

# Ubuntu 18
ravemanager::system_requirements("ubuntu", "18")

# Ubuntu 16
ravemanager::system_requirements("ubuntu", "16")

# =============== On Red Hat Enterprise Linux ===============

# Red Hat Enterprise Linux 8
ravemanager::system_requirements("redhat", "8")
# Red Hat Enterprise Linux 7
ravemanager::system_requirements("redhat", "7")

# =============== On CentOS ===============

# Red Hat Enterprise Linux 8
ravemanager::system_requirements("centos", "8")

# Red Hat Enterprise Linux 7
ravemanager::system_requirements("centos", "7")

# =============== On OpenSUSE ===============

# openSUSE 42.3
ravemanager::system_requirements("opensuse", "42")

# =============== On SUSE Linux Enterprise ===============

# SUSE Linux Enterprise 12.3
ravemanager::system_requirements("sle", "12")

}
```

`uninstall` *Uninstall RAVE components*

Description

Remove cache, python, and/or all settings. Please be aware that R, 'RStudio', and already installed R packages will not be uninstalled. Please carefully read printed messages.

Usage

```
uninstall(components = c("cache", "python", "all"))
```

Arguments

`components`	which component to remove, see example for choices.

Examples

```
if( FALSE ) {

  # remove cache only
  ravemanager::uninstall("cache")

  # remove python environment
  ravemanager::uninstall("python")
  # remove all sample data, settings files
  ravemanager::uninstall("all")

}
```

version_info *Print out 'RAVE' version information*

Description

Print out 'RAVE' version information

Usage

```
version_info(vanilla = FALSE)
```

Arguments

`vanilla`	whether to use vanilla packages in this function

Index