Package 'basilisk.utils'

July 18, 2025

Contents

---

binaryPath *Get binary paths*

Description

Get binary paths

Usage

```
condaBinary(loc)

pythonBinary(loc)
```

Arguments

loc   String containing the path to the root of a conda instance or environment.

Details

This code is largely copied from **reticulate**, and is only present here as they do not export these utilities for general consumption.

Value

String containing the path to the conda or Python executable inside `loc`. If `loc` is not supplied, the relative path from the root of a conda instance or environment is returned.

Author(s)

Aaron Lun

Examples

```
condaBinary()
pythonBinary()
```

---

configureEnvironments *Create environments with system installs*

Description

Create conda environments during installation of the R package, typically by calling this function in a package's `configure` file.

Usage

```
configureEnvironments(src)
```

Arguments

src   String containing the local path to an R source file that defines the environments
            to be created.

Details

Sometimes, the lazy creation of new environments is not desirable. For example, administrators
of multi-user R instances would not want to create a separate environment in each user's cache.
Similarly, users of Docker images would not want to recreate the environment inside every new
container.

In these cases, an alternative is to use a "system install", where the environments are created during
R package installation. Each environment is directly created in the package's installation direc-
tory and is immediately available when createEnvironment is called. Administrators can enable
this mode by setting the BIOCCMAKE_USE_SYSTEM_INSTALL environment variable to 1. Note that
this setting takes effect during R package installation so should be set before (re-)installation of
basilisk.utils and its dependencies.

Developers can support system installs by adding configure(.win) scripts to the root of their
package. These scripts should call configureEnvironments to trigger creation of the conda envi-
ronments during R package installation. Details of the environments to be created are taken from
the src file, which should be executable as a standalone R file (i.e., it can be sourced). Each conda
environment is defined as a list with a name ending in _args, where the list contains arguments to
createEnvironment.

Packages that support system installs should also set StagedInstall: no in their DESCRIPTION
files. This ensures that the conda environments are created with the correct hard-coded paths in the
package installation directory.

Value

NULL is invisibly returned. Conda environments are created in the R package installation directory
if system installs are enabled. Otherwise, no action is performed.

Author(s)

Aaron Lun

Examples

```
# If we have a package with an 'R/environments.R' file,
# we could put the following in our 'configure' file.
## Not run: configureEnvironments('R/environments.R')
```

---

createEnvironment   *Create a new conda environment*

Description

Create a new conda environment if it does not already exist in the cache/system.

Usage

```
createEnvironment(
  pkg,
  name,
  version,
  packages,
  cache.dir = NULL,
  ignore.cache = FALSE,
  conda = NULL,
  channels = "conda-forge",
  override.channels = TRUE,
  extra = "--quiet"
)
```

Arguments

pkg   String containing the name of the R package that owns this conda environment.

name   String containing the name of the environment.

version   String containing the version of the environment. Ignored for system installs.

packages   Character vector of conda packages (possibly with version specifications) to be
                  installed in this environment.

cache.dir   String containing the location of the cache for lazily instantiated environments.
                  If NULL, this defaults to a location returned by `R_user_dir` with package. Ig-
                  nored for system installs.

ignore.cache   Logical scalar indicating whether to ignore cached environments if they already
                  exist. Ignored for system installs.

conda   String containing the path to the conda command or executable. If NULL, a
                  suitable value is obtained from `find`.

channels   Character vector of channels to use for conda environment creation.
override.channels

                  Logical scalar indicating whether to set the `--override-channels` option dur-
                  ing conda environment creation.

extra   Character vector of additional arguments to pass to conda `create`.

Details

In general, createEnvironment should be called inside any function of a downstream package that
relies on the conda environment. On its first call, it will then lazily instantiate the environment
based on the specified arguments. All subsequent calls in the same or new R sessions will use the
cached environments.

The version string is used to distinguish between different versions of the same name environ-
ments. This allows package developers to safely update their environments without affecting other
R installations that are re-using the same cache. Older unused versions of the environment will be
automatically removed over time via **dir.expiry**.

Note that the version string does not necessarily have to be the same as the version of the pkg
package. Any version-like string is fine as long as they are compatible with `package_version`.
In fact, having independent versions for the environments and their parent package is often more
convenient, as it means that the environments don't always need to be recreated when the package
is updated.

To avoid lazy evaluation, administrators of an R installation can enable system installs, see [configureEnvironments](#) for details.

Value

String containing the path to the conda environment.

Author(s)

Aaron Lun

Examples

```
createEnvironment(
    pkg="basilisk.utils.test",
    name="test_env",
    version="1.0.0",
    packages="hdf5"
)

# Repeated calls will just get the same environment back.
createEnvironment(
    pkg="basilisk.utils.test",
    name="test_env",
    version="1.0.0",
    packages="hdf5"
)
```

---

defaults	*Defaults for* **biocconda**

Description

Defaults for **biocconda**

Usage

```
defaultCommand()

defaultDownloadVersion()

defaultMinimumVersion()

defaultCacheDirectory()
```

Details

The BIOCCONDA_CONDA_COMMAND environment variable will override the default setting of defaultCommand.

The BIOCCONDA_CONDA_DOWNLOAD_VERSION environment variable will override the default setting of defaultDownloadVersion.

The BIOCCONDA_CONDA_MINIMUM_VERSION environment variable will override the default setting of defaultMinimumVersion.

The BIOCCONDA_CONDA_CACHE_DIRECTORY environment variable will override the default setting of defaultCacheDirectory.

Value

For defaultCommand, a string specifying the expected command-line invocation of an existing conda installation.

For defaultDownloadVersion, a string specifying the version of conda to download if no existing installation can be found.

For defaultMinimumVersion, a string specifying the minimum version of an existing conda installation.

For defaultCacheDirectory, a string containing the path to the cache directory for **biocconda**-managed conda installations.

Author(s)

Aaron Lun

Examples

```
defaultCommand()
defaultDownloadVersion()
defaultMinimumVersion()
defaultCacheDirectory()
```

---

download *Install conda*

Description

Install conda via the Miniforge project to an appropriate destination path, skipping the installation if said path already exists.

Usage

```
download(
  download.version = defaultDownloadVersion(),
  cache.dir = defaultCacheDirectory(),
  ignore.cache = FALSE
)
```

Arguments

download.version
:   String specifying the Miniforge version to download.

cache.dir
:   String specifying the location of the directory in which to cache Miniforge installations.

ignore.cache
:   Logical scalar specifying whether to ignore any existing cached version of Miniforge, in which case the binaries will be downloaded again.

Details

This function was originally created from code in <https://github.com/hafen/rminiconda>, also borrowing code from **reticulate**'s install_miniconda for correct Windows installation. It downloads and runs a Miniforge installer to create a Bioconductor-managed Conda instance.

Whenever download is re-run, any old conda instances and their associated **basilisk** environments are deleted from the external installation directory. This avoids duplication of large conda instances after their obselescence.

Value

A conda instance is created at the cache location. Nothing is performed if a complete instance already exists at that location. A string is returned containing the path to the conda installation.

Author(s)

Aaron Lun

Examples

```
download()
```

---

find   *Find conda*

Description

Find an existing conda installation or, if none can be found, install a **biocconda**-managed conda instance.

Usage

```
find(
  command = defaultCommand(),
  minimum.version = defaultMinimumVersion(),
  can.download = TRUE,
  forget = FALSE,
  ...
)
```

Arguments

command	String containing the command to check for an existing installation.
minimum.version	
	String specifying the minimum acceptable version of an existing installation.
can.download	Logical scalar indicating whether to download conda if no acceptable existing installation can be found.
forget	Logical scalar indicating whether to forget the results of the last call.
...	Further arguments to pass to [download](#).

Details

If the `BIOCCONDA_FIND_OVERRIDE` environment variable is set to a command or path to a conda executable, it is returned directly and all other options are ignored.

By default, `find` will remember the result of its last call in the current R session, to avoid re-checking the versions, cache, etc. This can be disabled by setting `forget=TRUE` to force a re-check, e.g., to detect a new version of conda that was installed while the R session is active.

Value

String containing the command to use to run conda.

Author(s)

Aaron Lun

Examples

```
cmd <- find()
system2(cmd, "--version")
```

Index