The switchr Package

Gabriel Becker

June 2, 2015

Contents

1 Managing and populating package libraries

1.1 Creating and switching between package libraries

Different versions of R packages are appropriate for different applications, including package development, solo analysis, and collaboration. `switchr` provides a framework for creating, populating, managing, and switching between multiple package libraries from within an R session.

For safety within this vignette, we set the base `switchr` directory (the directory under which `switchr` libraries will be used) to a temporary directory.

```
> library(switchr)
> switchrBaseDir(file.path(tempdir(), ".switchr"))
> options(width=40, repos = c(CRAN="http://cran.rstudio.com"))
```

We create and use switchr package libraries by referring to them by name using the `switchTo()` function as seen below.

```
> switchTo("vign1")
```

When switching to a different package library, `switchr` displays messages indicating the (newly) active library and some basic information about it[1], which is of the form

```
[1] "Switched to the 'vign1' computing environment. 29 packages are currently available."
[2] "Packages installed in your site library ARE suppressed."
[3] "To switch back to your previous environment type switchBack()"
```

The first time this code is run a "vign1" library is created and activated. By activate we mean that it replaces the current User- and (by default) Site- library locations which R uses to find packages during the loading process.

To stop using "vign1" we simply call `switchBack()`:

```
> switchBack()
```

When switching to a different library, `switchr` attempts to unload all packages (other than switchr itself and its dependencies) in the currently loaded session. This generally works, but can occasionally lead to problems when R has trouble fully unloading a package. We also note here that R's design does not support the full unloading of S3 methods. This can lead to packages being spontaneously reloaded when the S3 generic is called with particular classes.

Users can indicate that packages should NOT be unloaded when switching libraries via the `switchrDontUnload` function which sets a package specific option for the remainder of the R session.

1.2 Removing package libraries

`switchr` package libraries can be removed via the `removeLib` function, like so

1.3 Representing package cohorts

Empty package libraries are, of course, of limited use. The other main functionality `switchr` offers is installation of packages from a wide variety of sources (git, svn, package repositories, and the CRAN Archive among them), both individually and as cohorts.

[1] The reason these messages are not displayed as output for other code-blocks in this document is that the information is conveyed via *messages*, which are not captured by the Sweave engine. We have chosen to use Sweave regardless due to its smaller dependency footprint.

We do this via a generalized form of a package repository called a *package manifest*, which includes a listing of packages and the information switchr needs to install them (primarily location of their source code).

We can create a package manifest from scratch via the `PkgManifest` function, though there convenience functions for common use-cases such as Github packages

```
> man = PkgManifest(name = "fastdigest",
+     url = "https://github.com/gmbecker/fastdigest",
+     type = "git")
> man

A package manifest (PkgManifest object)

Contains 1 packages and 5 dependency repositories

Packages:
        name type
1 fastdigest  git
```

The 'GithubManifest' convenience function builds manifests from GitHub username/repository pairs for package sources residing in the root directory of the master branch for the specified repository.

```
> man2 = GithubManifest("gmbecker/fastdigest",
+      "gmbecker/RCacheSuite")
> man2

A package manifest (PkgManifest object)

Contains 2 packages and 5 dependency repositories

Packages:
         name type
1  fastdigest  git
2 RCacheSuite  git
```

The 'GithubManifest' function accepts (a subset of) the same shorthand for locations within Github repositories as Wickham's `install_github` function from the `devtools` package. "/"s after the first indicate a subdirectories within the repository, and a "@@" indicates the (non-master) branch the manifest should point to. Package names that differ from the name of the repository are indicated via the argument names associated with the shorthand strings.

```
> man3 = GithubManifest(redland = "ropensci/redland-bindings/R/redland")
> man

A package manifest (PkgManifest object)

Contains 1 packages and 5 dependency repositories

Packages:
        name type
1 fastdigest  git

>
```

1.4 Representing and (re-)installing package cohorts/libraries

Beyond package manifests, `switchr` also supports *seeding manifests*, which extend package manifests by indicating a subset of the packages listed in the manifest, specific package-version information, or both.

```
> lman = libManifest()
> lman

A seeding manifest (SessionManifest object)

Describes a cohort of 159 package versions.
159 packages are listed in the underlying package manifest

Package versions:
    name            version
1   "switchr"       "0.9.3"
2   "AnnotationDbi" "1.30.1"
3   "BBmisc"        "1.9"
4   "BH"            "1.55.0-3"
... "..."           "..."
155 "stats4"        "3.2.0"
156 "survival"      "2.38-1"
157 "tcltk"         "3.2.0"
158 "tools"         "3.2.0"
159 "utils"         "3.2.0"
```

Package or seeding manifests can then be used to *seed* switchr libraries. During the new-library creation process, all packages listed in the seed - typically a *seeding manifest*, *package manifest*, or specially-purposed package repository - are installed. In the case that a *seeding manifest* is used, the exact, specified versions of the indicated packages are retrieved and installed, while for a package manifest the latest versions available are used.

We specify a seed via the *seed* argument to the `switchTo` function, like so:

```
> ## NOT RUN
> switchTo("vign2", seed = lman)
```

The above code will create a new switchr library called vign2 and populate it with the package versions listed in lman - i.e., the packages installed in the library used to build this vignette.

2 Installing specific packages and package versions

`switchr` also provides the `install_packages` function for installing specific packages or sets of packages from a variety of sources, including traditional repositories and package manifests. This function supports package dependencies, including non-repository to non-repository dependecies (e.g. between two or more packages on GitHub) on systems able to build packages from source.

```
> ## NOT RUN
> install_packages("RCacheSuite", man2)
```

We can also use `install_packages` to install specific versions of the desired packages.

```
> ## NOT RUN
> install_packages("fastdigest",
+                  versions = c(fastdigest= "0.5-0"),
+                  man = man2)
>
```

Note, however, that installing individual, non-current package versions without specifying versions for those packages' dependencies is likely to result in undefined package behavior, as the most current versions of the dependencies will be used. Packages will often fail to install entirely, though "successful" installation via this mechanism should not be taken as a guarantee that the packages will behave as expected.

2.1 Building consistent, historical package manifests

`switchr` does include experimental support for building a consistent manifest based on a specific version of a single package. This uses Csardi's the database of R package metadata developed for his `crandb` package.

The `cranPkgVersManifest` function accepts a (CRAN) package and a specific version. It then queries Csardi's database to determine versions of the pacakge's dependencies which were concurrent with the first or last day that the specified version was the current release on CRAN.

```
> oldman = cranPkgVersManifest(pkg = "randomForest", vers = "4.6-5")
> oldman

A package manifest (PkgManifest object)

Contains 3 packages and 0 dependency repositories

Packages:
          name    type
1 randomForest tarball
2 RColorBrewer tarball
3         MASS tarball
```

The resulting *package manifest*, while not a *seeding manifest*, points to source tarballs within the CRAN Web Archive corresponding to exact contemporary versions of the package and its depencies. Currently only packages published on CRAN are supported.

```
> manifest_df(oldman)$url

[1] "http://cran.r-project.org/src/contrib/Archive/randomForest/randomForest_4.6-5.tar.gz"
[2] "http://cran.r-project.org/src/contrib/Archive/RColorBrewer/RColorBrewer_1.0-5.tar.gz"
[3] "http://cran.r-project.org/src/contrib/Archive/MASS/MASS_7.3-16.tar.gz"
```

Users can also leverage Csardi's database to retreive a manifest representing the state of CRAN corresponding to the release of a particular R version, retroactively creating something similar to the repositories created by de Vries' `miniCRAN` and its successor, Revolution Analytic's `checkpoint`.

```
> oldman2 = rVersionManifest("3.1.1")
> oldman2

A package manifest (PkgManifest object)

Contains 4865 packages and 0 dependency repositories
```

```
Packages:
     name          type
1    "A3"          "tarball"
2    "abc"         "tarball"
3    "abcdeFBA"    "tarball"
4    "ABCExtremes" "tarball"
...  "..."         "..."
4861 "zoeppritz"   "tarball"
4862 "zoo"         "tarball"
4863 "zooimage"    "tarball"
4864 "zoom"        "tarball"
4865 "zyp"         "tarball"
```

This results a manifest of tarball locations, as above. It will contain locations in the currently-selected CRAN mirror and the CRAN Web Archive, depending on whether a particular package has been updated since the specified R version.

```
> head(manifest_df(oldman2)$url)

[1] "http://cran.rstudio.com/src/contrib/A3_0.9.2.tar.gz"
[2] "http://cran.r-project.org/src/contrib/Archive/abc/abc_1.8.tar.gz"
[3] "http://cran.rstudio.com/src/contrib/abcdeFBA_0.4.tar.gz"
[4] "http://cran.rstudio.com/src/contrib/ABCExtremes_1.0.tar.gz"
[5] "http://cran.rstudio.com/src/contrib/ABCoptim_0.13.11.tar.gz"
[6] "http://cran.rstudio.com/src/contrib/ABCp2_1.1.tar.gz"
```

3 Using switchr in dynamic documents

While we chose not to depend on Xie's `knitr` package, `switchr` can be used to enhance reproducibility of analyses residing in dynamic documents which are processed with that system, e.g. .Rmd files. To do this, users need simply ensure that knitr and its dependencies are not unloaded during the library switching process. This is achieved via the `switchrDontUnload` function, which sets a persistent option telling `switchr` to ignore those packages when emptying the current R session. Base packages, `switchr` itself, and its dependencies are always ignored in this way.

```
> ## NOT RUN
> switchrDontUnload(c("knitr", "evaluate", "digest", "formatR", "highr",
+                     "markdown", "stringr"))
>
```

In the knitr case, for example, this function can be called from the outer R session or — preferably — from an early, intialization code chunk within the dynamic document itself.