formatR: Format R Code Automatically

Yihui Xie*

October 26, 2011

The package **formatR** (Xie, 2011) was designed to help users tidy (reformat) their source code. This vignette serves as a showcase of the function *tidy.source()*, and a broader introduction can be found in `https://github.com/yihui/formatR/wiki/`.

1 The workhorse *tidy.source()*

The main function in this package is *tidy.source()*, which can take a file as input, parse it and write the formatted code to the console or a file.

```r
> library(formatR)
> usage(tidy.source)

tidy.source(source = "clipboard", keep.comment, keep.blank.line,
    keep.space, replace.assign, output = TRUE, text = NULL, width.cutoff = 0.75 *
        getOption("width"), ...)
```

There are four options which can affect the final output: `keep.comment`, `keep.blank.line`, `keep.space` and `replace.assign`. They are explained in the help page; see `?tidy.source`. For example, if we do not want to keep the blank lines in the code, we can first specify a global option like this:

```r
> options(keep.blank.line = TRUE)  # not really need to do so; default is TRUE
```

The option `width` will affect the width of the output, e.g. we can specify a narrow width:

```r
> options(width = 85)
```

Here are some examples taken from the help page:

```r
> library(formatR)
> ## use the 'text' argument
> src = c("    # a single line of comments is preserved", "1+1", "if(TRUE){",
+     "x=1  # inline comments", "}else{", "x=2;print('Oh no... ask the right bracket to
go away!')}",
+     "1*3 # one space before this comment will become two!", "2+2+2    # 'short comments'",
+     "   ", "lm(y~x1+x2)  ### only 'single quotes' are allowed in comments",
+     "\t\t## tabs/spaces before comments: use keep.space=TRUE to keep them",
+     "'a character string with \t in it'", "# note tabs will be converted to spaces when
keep.space=TRUE",
+     "1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1  ## comments after a long line",
+     paste("## here is a", paste(rep("long", 20), collapse = " "),
+         "comment"))
```

---

*Department of Statistics, Iowa State University. Email: xie@yihui.name

Source code

```
> cat(src, sep = "\n")

    # a single line of comments is preserved
1+1
if(TRUE){
x=1  # inline comments
}else{
x=2;print('Oh no... ask the right bracket to go away!')}
1*3 # one space before this comment will become two!
2+2+2    # 'short comments'

lm(y~x1+x2)  ### only 'single quotes' are allowed in comments
                ## tabs/spaces before comments: use keep.space=TRUE to keep them
'a character string with          in it'
# note tabs will be converted to spaces when keep.space=TRUE
1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1  ## comments after a long line
## here is a long long long long long long long long long long long long long long long long long lon
```

Format and replace = with <-

```
> tidy.source(text = src, replace.assign = TRUE)

# a single line of comments is preserved
1 + 1
if (TRUE) {
    x <- 1  # inline comments
} else {
    x <- 2
    print("Oh no... ask the right bracket to go away!")
}
1 * 3  # one space before this comment will become two!
2 + 2 + 2  # 'short comments'

lm(y ~ x1 + x2)  ### only 'single quotes' are allowed in comments
## tabs/spaces before comments: use keep.space=TRUE to keep
#    them
"a character string with \t in it"
# note tabs will be converted to spaces when keep.space=TRUE
1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 +
    1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1  ## comments after a long line
## here is a long long long long long long long long long long
#    long long long long long long long long long long comment
```

Preserve leading spaces

```
> tidy.source(text = src, keep.space = TRUE)

    # a single line of comments is preserved
1 + 1
if (TRUE) {
    x = 1  # inline comments
```

```
} else {
    x = 2
    print("Oh no... ask the right bracket to go away!")
}
1 * 3  # one space before this comment will become two!
2 + 2 + 2  # 'short comments'

lm(y ~ x1 + x2)  ### only 'single quotes' are allowed in comments
                ## tabs/spaces before comments: use keep.space=TRUE to keep them
"a character string with          in it"
# note tabs will be converted to spaces when keep.space=TRUE
1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 +
    1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1  ## comments after a long line
## here is a long long long long long long long long long long long long long long long long long lon
```

Discard blank lines

```
> ## note the 11th line (was a blank line)!
> tidy.source(text = src, keep.blank.line = FALSE)

# a single line of comments is preserved
1 + 1
if (TRUE) {
    x = 1  # inline comments
} else {
    x = 2
    print("Oh no... ask the right bracket to go away!")
}
1 * 3  # one space before this comment will become two!
2 + 2 + 2  # 'short comments'
lm(y ~ x1 + x2)  ### only 'single quotes' are allowed in comments
## tabs/spaces before comments: use keep.space=TRUE to keep
#    them
"a character string with \t in it"
# note tabs will be converted to spaces when keep.space=TRUE
1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 +
    1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1  ## comments after a long line
## here is a long long long long long long long long long long
#    long long long long long long long long long long comment
```

Discard comments

```
> tidy.source(text = src, keep.comment = FALSE)

1 + 1
if (TRUE) {
    x = 1
} else {
    x = 2
    print("Oh no... ask the right bracket to go away!")
}
1 * 3
```

```
2 + 2 + 2
lm(y ~ x1 + x2)
"a character string with \t in it"
1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 +
    1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1
```

2   Applications

This package has been used in a few other R packages. For example, **Rd2roxygen** (Wickham and Xie, 2011) uses **formatR** to reformat the code in the usage and examples sections in Rd files, since the code generated by **roxygen2** is not well-formatted; **pgfSweave** (Bracken and Sharpsteen, 2011) can tidy the Sweave code chunks when the Sweave option `tidy` is `TRUE` (just like the code in this vignette).

About this vignette

You might be curious about how this vignette was generated, because it looks different from other Sweave-based vignettes. The answer is **pgfSweave** (Bracken and Sharpsteen, 2011). The real vignette is in L$_Y$X, which can be found here:

```
> system.file("doc", "formatR.lyx", package = "formatR")
```

Read this blog entry for details and how to reproduce the vignette: `http://yihui.name/en/?p=602`.

References

Bracken C, Sharpsteen C (2011). *pgfSweave: Quality speedy graphics compilation with Sweave*. R package version 1.2.1, URL `http://CRAN.R-project.org/package=pgfSweave`.

Wickham H, Xie Y (2011). *Rd2roxygen: Convert Rd to roxygen documentation*. R package version 1.0-7, URL `https://github.com/yihui/Rd2roxygen`.

Xie Y (2011). *formatR: Format R Code Automatically*. R package version 0.3-2, URL `http://CRAN.R-project.org/package=formatR`.