Roxygen Tutorial

Peter C. Danenberg[1] and Manuel J. A. Eugster[2]

[1]pcd@roxygen.org
[2]manuel.eugster@stat.uni-muenchen.de

useR! 2010, July 20th 2010

Roadmap

1. Introduction: Literate Programming and Roxygen
2. Exemplar package: Monte Carlo π approximation
3. Use Roxygen:
   From one single source file to a package which passes `R CMD check`.
4. Exemplar roclet: BibTex references
5. Extend Roxygen:
   From special comment blocks to valid R package standard outcomes.
6. Summary and Future of Roxygen
7. Questions

Part I

Introduction

Literate programming

"When was the last time you spent a pleasant evening in a comfortable chair, reading a good program?"
— Bentley (1986)

"I believe that the time is ripe for significantly better documentation of programs, and that we can best achieve this by considering programs to be works of literature."
— Knuth (1992)

Literate programming

Literate programming: Interleaving code and documentation chunks with *weave* and *tangle*; e.g. Sweave for R (Leisch, 2002).

Interface documentation: Documentation statements as comments; e.g. Doxygen for C/C++ (van Heesch, 2008) and Javadoc for Java (Sun Microsystems, Inc., 2008).

Roxygen

Roxygen enables **in-source** specification of

- documentation and
- package related information.

Google Summer of Code 2008 project by Peter Danenberg, mentored by Manuel J. A. Eugster.

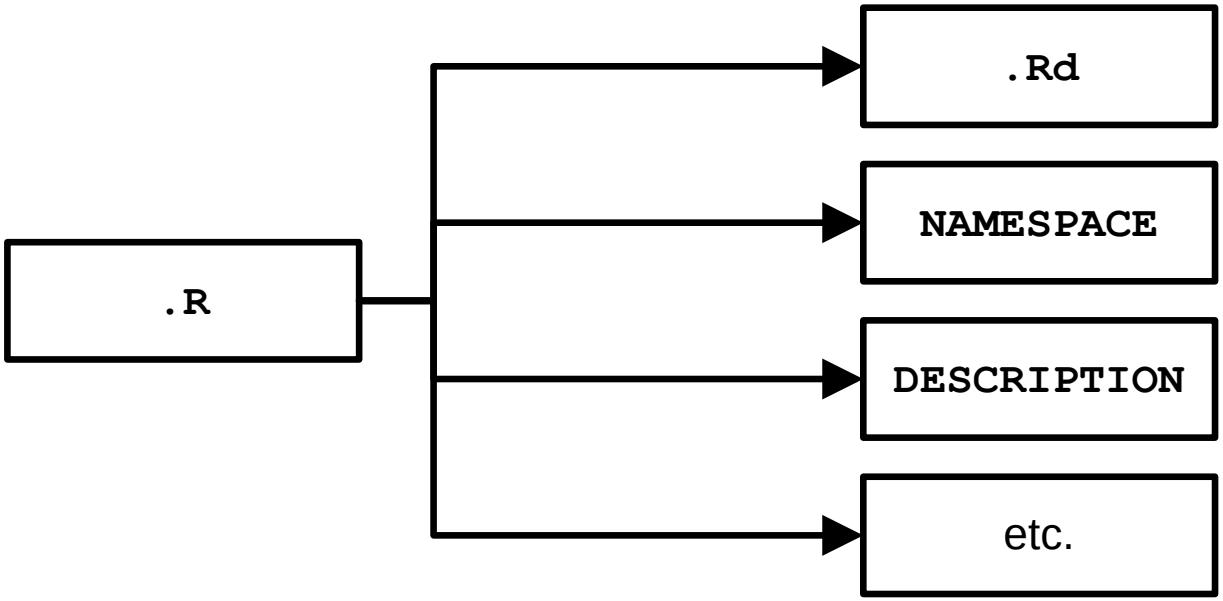
Concept of Roxygen

Roxygen processes special **documentation blocks** in front of R statements. The blocks are made up of two parts – a **textual description** and **descriptive tags**.

```
#' Description
#'
#' Details
#'
#' @param a Description
#' @param b Description
#' ...
f <- function(a, b) {
```

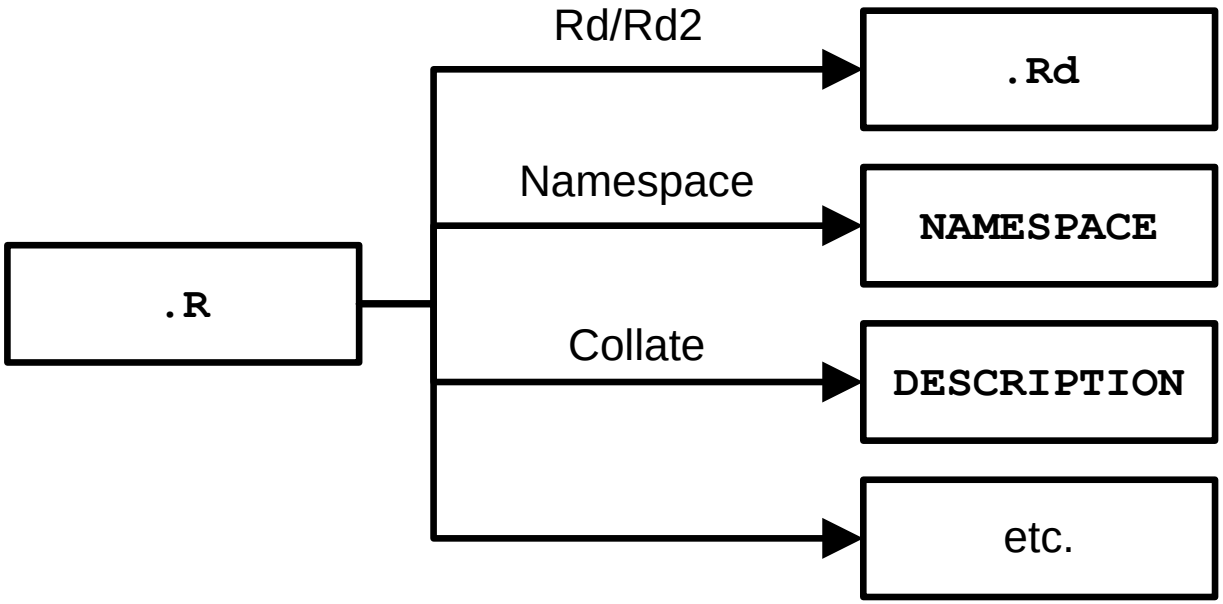
Concept of Roxygen

The programmer writes all **information into the R source files** and Roxygen writes them to the adequate places.



Concept of Roxygen

The real work is done by **Roclets**. Each Roclet understands a set of tags and processes them to some outcome.



Roxygen at the time

Available Roclets are:

`Rd`	Processes Rd tags and and generates Rd help files.
`Rd2`	Another implementation with, for example, experimental (basic) S4 support and merging of different Rd files.
`namespace`	Processes namespace directives tags (import/export); generates a `NAMESPACE` file.
`collate`	Processes `@includes` and merges with the `Collate` field in a pre-existing `DESCRIPTION` file.
`callgraph`	Procudes a static call graph from a given function at a given depth with or without primitives.

Find the provided tags with `?make.`*`rocletname`*`.roclet`.

Roxygen at the time

Chosen packages which make use of Roxygen:

roxygen	Literate Programming in R (CRAN, R-Forge).
archetypes	Archetypal Analysis (CRAN, R-Forge).
benchmark	Benchmark Experiments Toolbox (CRAN, R-Forge).
mlr	Machine Learning in R (R-Forge).
profr	An alternative display for profiling information (CRAN).

Roxygen at the time

Package:

stable	`http://CRAN.R-Project.org/package=roxygen`
devel	`http://R-Forge.R-Project.org/projects/roxygen`

Communication:

Website	`http://roxygen.org`
Mailing list	`roxygen-devel@lists.r-forge.r-project.org`
IRC	`#roxygen` on Freenode

Tutorial:

Material	`http://roxygen.org/tutorial`

Part II

Exemplar package: Monte Carlo π approximation

Monte Carlo π approximation

If a circle of radius R is inscribed inside a square with side length $2 * R$, then the area of the circle will be $\pi * R^2$ and the area of the square will be $(2 * R)^2$. So the ratio of the area of the circle to the area of the square will be $\frac{\pi}{4}$.

This means that, if you pick n points at random inside the square, approximately $n * \frac{\pi}{4}$ of those points should fall inside the circle.

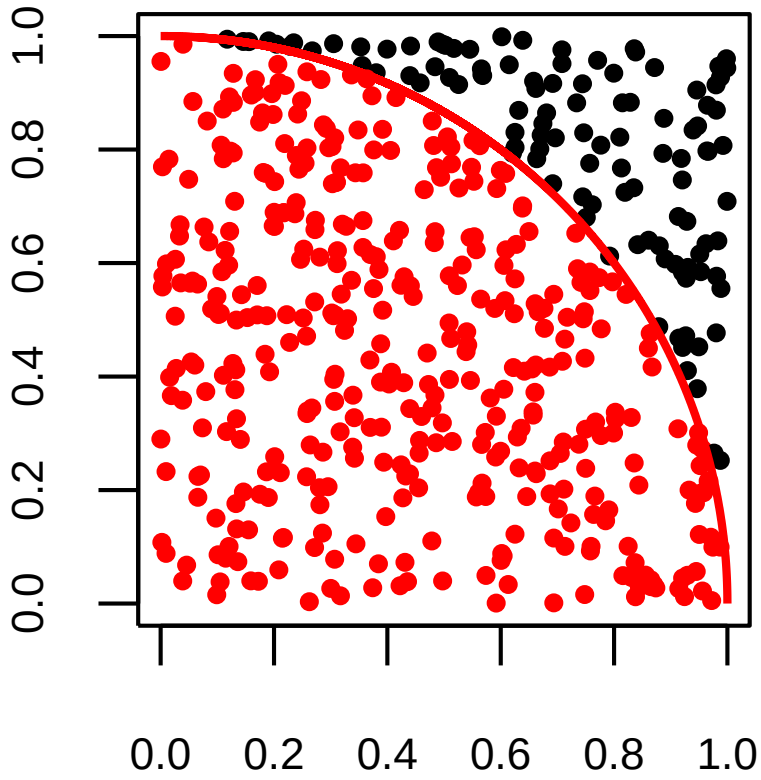
(*) Andersson (2010)

R implementation

```
> x <- mcpi(500)
> x

3.104

> summary(x)

3.104 (estimated by 388 hits from 500 trials)
```

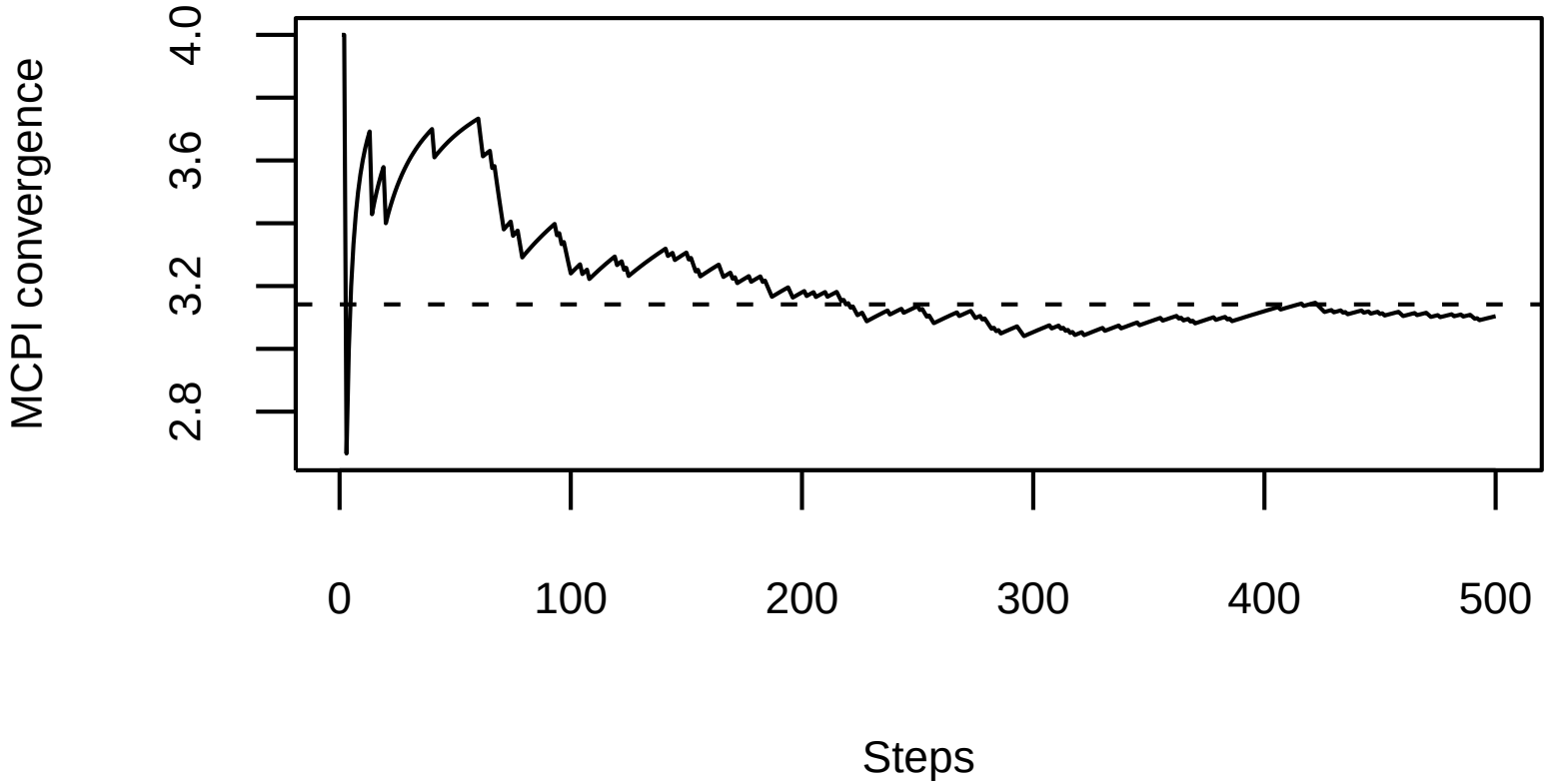
R implementation

```
> plot(x)
```

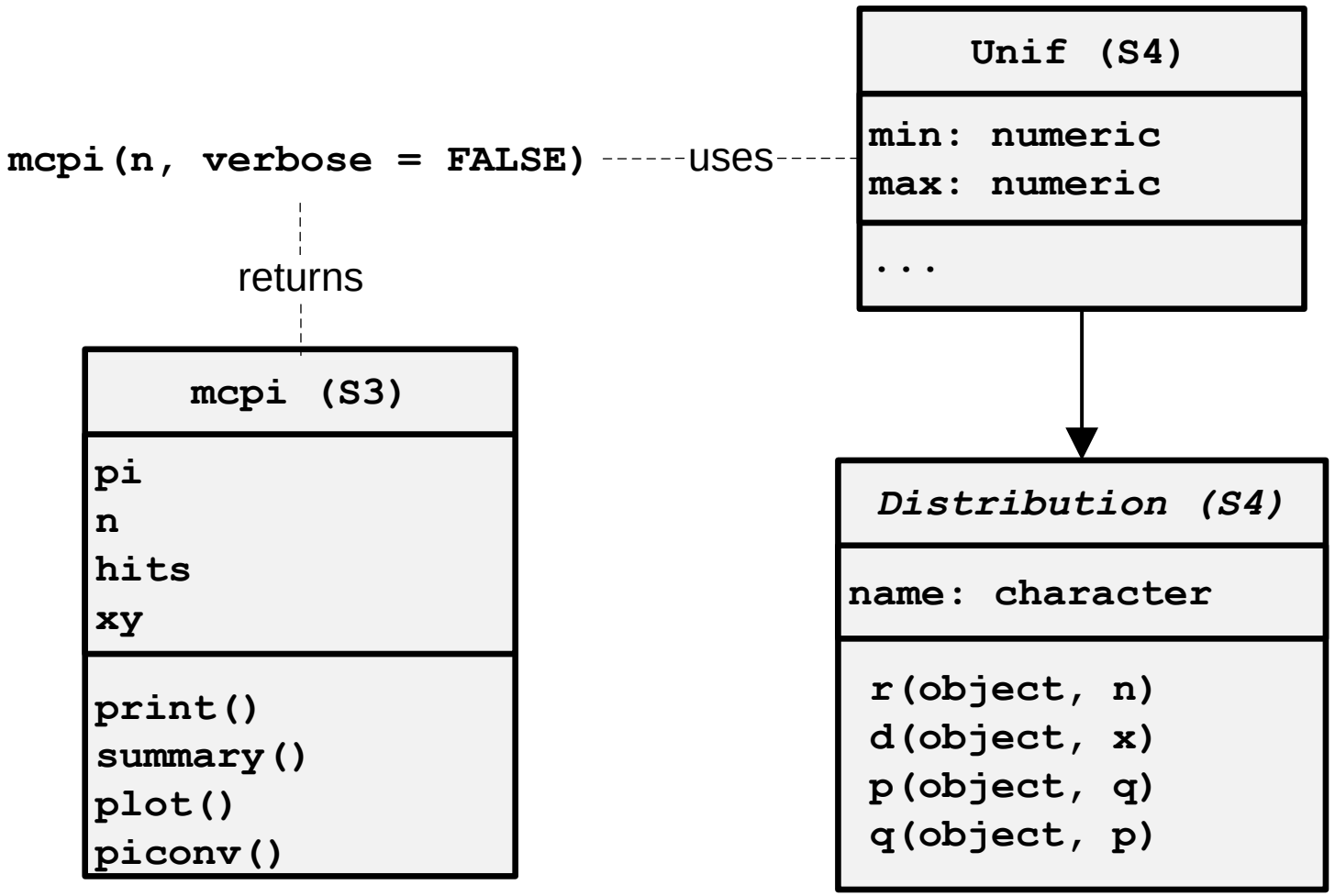


R implementation

```
> piconv(x)
```



R implementation details



⇒ Live demonstration.

Part III

Use Roxygen

Roadmap

From one single R source code file to a package which passes `R CMD check`:

1. Divide into multiple R source code files based on related content; use the `collate` roclet.
2. Extend R source code files with in-source documentation; use the `Rd` and the `Rd2` roclet.
3. Create a package namespace; use the `namespace` roclet.

Execute Roxygen

Process a package with the `Rd`/`Rd2`, `namespace` and `collate` roclets.

Within R:

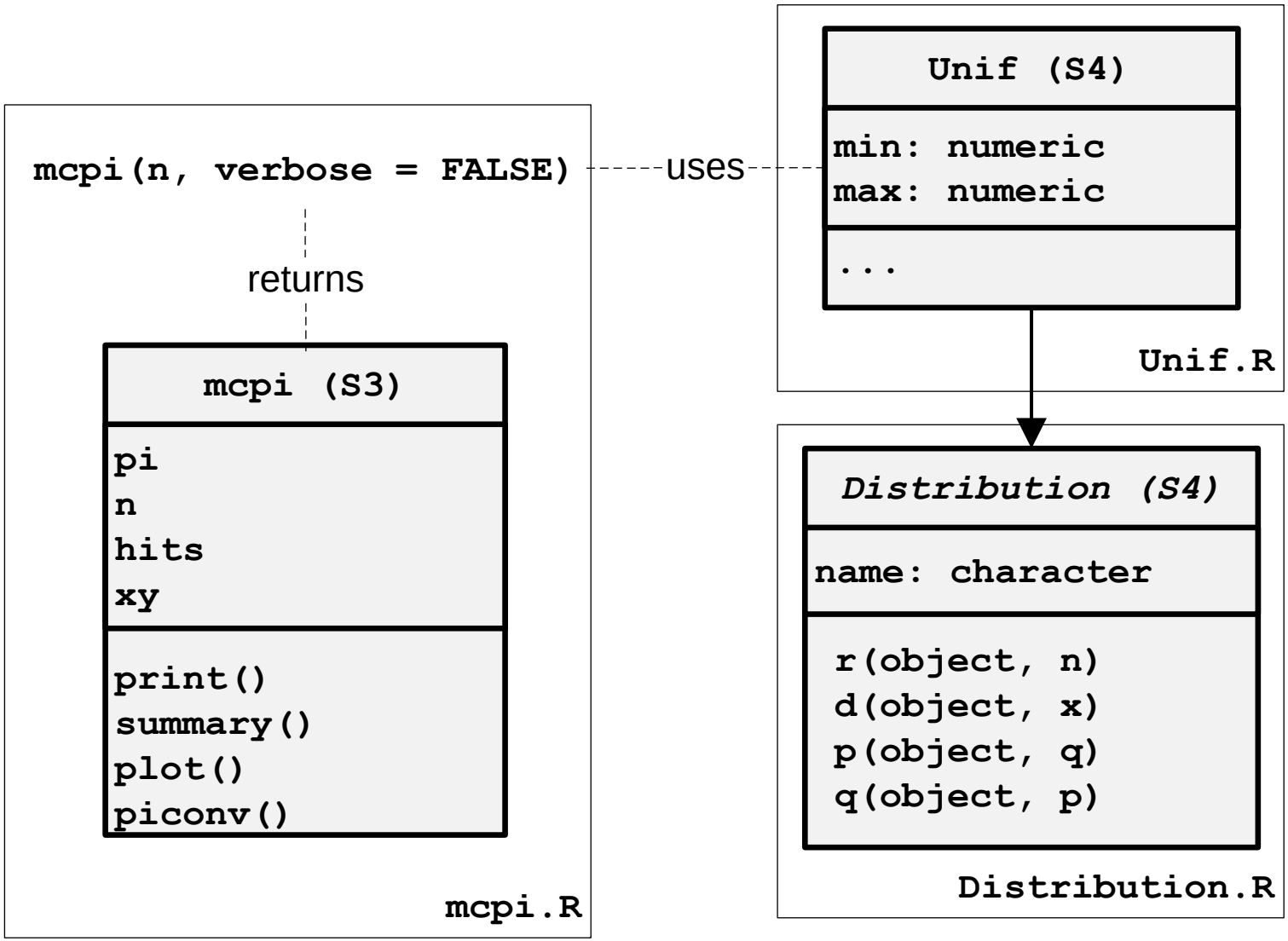
```
> library(roxygen)
> roxygenize("mcpi")
```

Command line:

```
$ R CMD roxygen mcpi
```

⇒ Live demonstration.

In-source "collate" definition



In-source "collate" definition

The **collate** roclet provides an include directive for source files to specify dependencies. It topologically sorts the dependencies and writes the `Collate` field in the `DESCRIPTION` file.

R/mcpi.R

```
#' @include Unif.R
{}

...
```

See `?make.collate.roclet`.

⇒ Live demonstration.

In-source documentation

The **Rd** roclet is the original implementation supporting functions and S3 constructs.

The **Rd2** roclet is a new implementation using the `Rd` structure defined by `parse_Rd()` (`R` $\geq$ 2.9). It additionally supports basic S4 constructs documentation.

Note that in-source documentation reduces the flexibility of writing Rd files, but enables standardized documentation.

See `?make.Rd.roclet` and `?make.Rd2.roclet`.

$\Rightarrow$ Live demonstration.

In-source documentation

Functions

```
#' Monte-Carlo PI approximation.
#'
#' If a circle of radius \eqn{R} is inscribed inside ...
#'
#' @param n Number of trials
#' @param verbose Print information during execution
#' @return S3 \code{mcpi} object; a list consisting of
#'   \item{pi}{the approximation of pi}
#'   \item{n}{the number of trials}
#'   \item{hits}{the number of hits}
#' @example mcpi/sandbox/mcpi.R
#' @references \url{http://www.eveandersson.com/pi/...}
#' ...
mcpi <- function(n, verbose=FALSE) {
```

$\Rightarrow$ Live demonstration.

In-source documentation

Functions, static documentation

Base documentation file for "static documentation"; file content is merged with documentation computed by Roxygen.

man/mcpi.Rd

```
\title{Monte-Carlo PI approximation}
\description{
  An implementation of the Monte-Carlo PI approximation.
}
\details{
  If a circle of radius \eqn{R} is inscribed inside ...
}
\references{\url{http://www.eveandersson.com/pi/...}}
```

⇒ Live demonstration.

In-source documentation

S3 methods

```
#' @param x A \code{mcpi} object
#' @param ... Ignored
#' @method print mcpi
#' @nord
#' ...
print.mcpi <- function(x, ...) {

#' @param object A \code{mcpi} object
#' @param ... Ignored
#' @method summary mcpi
#' @rdname mcpi
#' ...
summary.mcpi <- function(object, ...) {
```

$\Rightarrow$ Live demonstration.

In-source documentation

S3 generics and their methods

```
#' Visualizing the convergence of PI.
#' @param x An object
#' @param ... Further arguments
piconv <- function(x, ...) {

#' \code{piconv} implementation for \code{mcpi} objects.
#'
#' Visualizing the convergence of the Monte-Carlo PI approximation.
#' ...
#' @rdname piconv
piconv.mcpi <- function(x, xlab = "Steps",
                        ylab = "MCPI convergence", ...) {
```

⇒ Live demonstration.

In-source documentation

S4 classes

```
#' ...
#' @slot min Lower limit of the distribution
#' @slot max Upper limit of the distribution
setClass("Unif",
         contains = c("Distribution"),
         representation = representation(
           min = "numeric",
           max = "numeric"),
         prototype = prototype(
           name = "Uniform distribution",
           min = 0,
           max = 1))
```

Note that Roxygen needs a "complete" class specification.

$\Rightarrow$ Live demonstration.

In-source documentation

S4 generics

```
#' Random deviates.
#' @param object An object
#' @param n Number of random numbers
setGeneric("r",
function(object, n, ...) {
  standardGeneric("r")
})
```

In-source documentation

S4 methods

```
#' Random deviates.
#' @param object \code{\linkS4class{Unif}} object
#' @param n Number of random numbers
#' @rdname r-methods
setMethod("r",
          signature = signature(object = "Unif", n = "numeric"),
          function(object, n) {
```

⇒ Live demonstration.

In-source documentation

Further possibilities

Documentation of deprecated functions; set

```
#' ...
#' @rdname pkg-deprecated
depricatedfn <- function(x, ...) {
```

Package description:

```
#' This package ...
#' @rdname pkg-package
{}
```

In-source documentation

Odds and ends

- Reduces the flexibility of writing Rd files, but enables standardized documentation.
- The new Rd structure defined by `parse_Rd()` allows the development of an Rd API and might give back some of the flexibility.
- With the new help system (since R 2.10) a lot of things with S4 will become easier; e.g., finding all methods for a specific class.

In-source "NAMESPACE" definition

The **namespace** roclet enables export, import and `useDynLib` directives.

See `?make.namespace.roclet`.

In-source "NAMESPACE" definition

Export and import

```
#' ...
#' @export
mcpi <- function(n, verbose = FALSE) {

#' ...
#' @S3method print mcpi
print.mcpi <- function(x, ...) {
```

In-source "NAMESPACE" definition

Export and import

```
#' ...
#' @exportClass Unif
setClass("Unif",

#' ...
#' @exportMethod r
setGeneric("r", function(object, n, ...) {

#' ...
#' @importFrom stats runif
setMethod("r",
          signature = signature(object = "Unif", n = "numeric"),
```

⇒ Live demonstration.

R CMD check ... ?

```
> library("roxygen")
> options(useFancyQuotes = FALSE)
> roxygenize("mcpi", unlink.target = TRUE, use.Rd2 = TRUE)

$ R CMD build mcpi.roxygen
$ R CMD check mcpi_0.1.tar.gz
```

$\Rightarrow$ Live demonstration.

Part IV

Exemplar roclet: BibTex

BibTex roclet

The roclet provides the two tags `@bibliography` and `@cite`:

`@bibliograph`: defines the `.bib` file; default is `REFERENCES.bib` within the package directory.

`@cite`: references a citation by its key.

BibTex roclet

- Preprocessor for the Rd/Rd2 roclets.
- BibTex file parsing is done by the `bibtex` package; e.g.,

```
citationList <- read.bib(file = "REFERENCES.bib")
```

which returns an object `citationList` with `citation` elements – the common citation objects in R; see `?citation` and `?citeEntry`.

Exemplar citation

```
#' ...
#' @references
#'   @bibliography roxygen.bib
#'   @cite Leisch2002
f <- function(...) {
```

is replaced by

```
#' @references
#'   Friedrich Leisch. Sweave: Dynamic Generation of
#'   Statistical Reports Using Literate Data Analysis.
#'   Compstat 2002 --- Proceedings in Computational
#'   Statistics, pages 575--580. Physica Verlag,
#'   Heidelberg, 2002.
f <- function(...) {
```

Part V

Extend Roxygen

Roadmap

From special comment blocks to valid R package standard outcomes:

1. The Roxygen parser
2. Roclets
   - Parsing association lists (`prerefs`)
   - Parsing expression trees (`srcrefs`)
   - Processing the parse tree

Part VI

Summary and the Future

Roxygen in a nutshell

All information is written into the R source files:

- A Roxygen documentation block is prefaced with `#'`, and
- is made up of a textual description and descriptive tags.

Roclets understand a set of tags and process them to some outcome:

- Available roclets are, for example, the `Rd` and `namespace` roclets.
- It is easy to implement an individual roclet.

Use Roxygen:

- Within R: `> roxygenize("...")`.
- Command line: `$ R CMD roxygen ...`

Planned future features

- Complete S4 integration and integration of other object systems (like `proto`).
- Full usage of the new Rd structure.
- Generation of a "programmers documentation" (like Doxygen and Javadoc).
- Support of Roxygen on R-Forge; reasonable ESS support; a roclet library.

Part VII

Appendix

References

Eve Andersson. Calculation of pi using the monte carlo method. Website, 2010. Available online at `http://www.eveandersson.com/pi/monte-carlo-circle`; visited on July 8th 2010.

Jon Bentley. *Communications of the ACM*, 1986.

Donald E. Knuth. *Literate Programming*. CSLI, 1992. ISBN 0-937073-80-6. CSLI Lecture Notes, No. 27.

Friedrich Leisch. Sweave: Dynamic generation of statistical reports using literate data analysis. In Wolfgang Härdle and Bernd Rönz, editors, *Compstat 2002 — Proceedings in Computational Statistics*, pages 575–580. Physica Verlag, Heidelberg, 2002. URL `http://www.stat.uni-muenchen.de/~leisch/Sweave`. ISBN 3-7908-1517-9.

Sun Microsystems, Inc. *Javadoc*, 2008. URL `http://java.sun.com/j2se/javadoc/`.

Dimitri van Heesch. *Doxygen – Source code documentation generator tool*, 2008. URL `http://www.stack.nl/~dimitri/doxygen/`.