Package: flippingtables (via r-universe)

August 31, 2024

Title Override and cycle between table print methods

Version 0.0.3

Description User can configure a list of alternative print methods and table classes to which they apply. Facilitates cycling between print methods with a keyboard shortcut.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Imports crayon, glue, magrittr, rlang, stats, utils

Suggests palmerpenguins, pillar, testthat (>= 3.0.0)

Config/testthat/edition 3

Repository https://milesmcbain.r-universe.dev

RemoteUrl https://github.com/milesmcbain/flippingtables

RemoteRef main

RemoteSha 0e24d9e5c4da68579f3a7e601394f7356c71136f

Contents

default_print *Use the default print method for the table class*

Description

This function will dispatch the default print method for the table, if any of the table's classes have been registered for flipping. Where the table has multiple classes registered, the priority is given to the order specified in the registration config.

Usage

```
default_print(x, ..., .args = NULL)

print_arg(arg_aliases, arg_value)
```

Arguments

x	the table to print
...	args forwarded to the default print function
.args	configured args forwarded to the default print function using special argument matching rules, avoiding partial argument matching. See Details.
arg_aliases	A vector of aliases to use for this argument. The first alias matching a formal argument of the default print function will be used for the name of arg_value.
arg_value	The value to be bound to one of the arg_aliases that matches a formal argument of the default print function.

Details

Use this in printer_fns configuration in register_flips(). Do not use print() as that creates an unavoidable S3 dispatch loop.

Since this function stands in for many functions simultaneously, issues with argument routing and partial matching can arise when forwarding arguments with ....

An example would be default_print(x, n = 100), targeting 100 rows for class tbl. When the n arg is forwarded to base::print.data.frame instead of pillar::print.tbl it ends up getting matched to na.print due to partial argument matching, and thus causes an error.

.args is provided as a mechanism to work around the argument partial matching problem. If configured, it must be a list of print_arg() calls which are pairs of argument name alias vectors and a value. For each entry in .args the first alias that matches a formal argument in the print function to be called is used as the name paried with the corresponding value. If there is no match, that value is not passed. This means that partial argument matching cannot occur, and that it is possible to account for the same concept being represented by different argument names in different print functions.

Examples

```
## Not run:
# This example demonstrates the use of .args. Arguments can still be
# forwarded to all default print methods using `...` if that would not create
# problems - there's no harm in trying and seeing that simpler method first.
register_flips(
  printer_fns = list(
    function(x) default_print(x, .args = list(
      print_arg(c("n", "max"), 1),
      # Here 'n' is matched ahead of 'max',  so if n matches a formal arg exactly
      # nothing will be passed for max.
      print_arg(c("width"), 40)
    )),
    function(x) default_print(x, .args = list(
      print_arg(c("n"), 2),
      print_arg(c("max"), 20),
      # Here 'n' and 'max' are matched independently, both, either, or neither
      # will be passed depending on whether they match a formal arg exactly
      print_arg(c("width"), 40)
    ))
  ),
  printed_classes = list(
    print_override(class = "tbl", pkg_namespace = "pillar"),
    print_override(class = "data.frame", pkg_namespace = "base")
  )
)

## End(Not run)
```

`flip` *Cycle through configured print methods*

Description

‘print()“ methods are kept in the order supplied in `register_flips()`, calling this function cycles between them changing the active print method.

Usage

```
flip(last_value = .Last.value)
```

Arguments

`last_value`	The value to be printed with the next active print method cycled to by `flip()` defaults to `.Last.value`. Mainly used for testing.

Details

if the `.Last.value` hs a class in `printed_classes` supplied to `[register_flips()]`, it is reprinted using the current print method when this function is called.

---

flip_off — *Disable cycling between print methods with flip()*

Description

This undoes the print method rerouting done by `flip_on()`. Print methods are returned to defaults for all configured classes. The configuration created by `register_flips()` remains and can be enabled again using `flip_on()`.

Usage

```
flip_off()
```

---

flip_on — *Enable cycling between print methods with flip()*

Description

When a configuration has been created with `register_flips()` this function uses that config to reroute the print methods of the configured classes, enabling printing with user-defined methods and cycling between them.

Usage

```
flip_on()
```

---

print_override — *Create an object class - package namespace pair for print override*

Description

The combination of class and namespace is used to determine the name of the print method that is to be rerouted (overidden). E.g. `class = "data.frame"` and `pkg_namespace = "base"` leads to a call like `utils::getFromNamespace(paste0("print.", class), pkg_namespace)` to get `base::print.data.frame`.

Usage

```
print_override(class, pkg_namespace)
```

Arguments

class	the class of the object to override the print for
pkg_namespace	the package namespace where the print method can be found

---

register_flips *Register alternative print methods and classes to which they apply*

Description

This function configures the flipping between `print()` methods in the current sesson, but does not enable it. `flip_on()` and `flip_off()` enable and disable flipping between configured `print()` methods for configured classes.

Usage

```
register_flips(printer_fns, printed_classes)
```

Arguments

printer_fns	a list of functions to be cycled between in order supplied.
printed_classes	a lit of class / package namespace pairs, each created with a call to `print_override()`. The print method to override is assumed to be `<pkg_namespace>:::<class>.print()`.

Details

The order of `printed_classes` listed as `print_override()` calls is somewhat important if `default_print()`is amongprint which is always used as the last resort since most other table classes are a specialisation of it.

The list of `printer_fns` may contan named items. If the print method has a name, a message will be printed containing the name when the method if flipped to.

Examples

```
## Not run:
library(flippingtables)
register_flips(
  printer_fns = list(
   paint::paint,
   function(x) withr::with_options(list(width = 300), default_print(x)),
   function(x) withr::with_options(list(width = 40), default_print(x, n = 50))),
  printed_classes = list(
   print_override(class = "tbl", pkg_namespace = "pillar"),
   print_override(class = "data.frame", pkg_namespace = "base"),
   print_override(class = "data.table", pkg_namespace = "data.table")
  )
 )
flip_on() # now the configuration is live and can be used with flip()

## End(Not run)
```

Index