# Package 'VanillaCalendar'

August 28, 2026

**Title** Interactive Calendar and Date Picker Widget

**Version** 1.0.0

**Description** An R interface to the 'Vanilla Calendar Pro' 'JavaScript' library (version 3.2.0, bundled) for creating interactive and customizable calendar widgets and date pickers. The package enables integration of modern calendar components in R and 'Shiny' applications, with support for date, month, year and time selection, popup input mode, theming, and in-place updates from the 'Shiny' server.

**URL** https://github.com/ESCRI11/vanilla-calendar-r

**BugReports** https://github.com/ESCRI11/vanilla-calendar-r/issues

**License** GPL (>= 3)

**Copyright** The bundled Vanilla Calendar Pro library is copyright Yury Uvarov and released under the MIT licence; see inst/htmlwidgets/lib/VanillaCalendar-3.2.0/LICENSE.

**Encoding** UTF-8

**Config/roxygen2/version** 8.1.0

**Imports** htmlwidgets

**Suggests** shiny, testthat (>= 3.0.0), shinytest2, chromote, htmltools, withr, knitr, rmarkdown

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Xavier Escribà Montagut [aut, cre], Yury Uvarov [ctb, cph] (Author of the bundled 'Vanilla Calendar Pro' library)

**Maintainer** Xavier Escribà Montagut <xavier.escriba.montagut@gmail.com>

**Repository** CRAN

**Date/Publication** 2026-08-28 10:00:15 UTC

# Contents



---

`VanillaCalendar` *Vanilla Calendar Widget*

---

## Description

Creates a Vanilla Calendar Pro calendar or date picker for use in R, R Markdown and Shiny applications.

## Usage

```
VanillaCalendar(
  options = list(),
  width = NULL,
  height = NULL,
  elementId = NULL
)
```

## Arguments

| | |
|---|---|
| `options` | A named list of Vanilla Calendar Pro options, passed to the JavaScript library. Every option documented in the upstream reference is accepted; names are validated against the bundled library version and unknown names raise a warning. `Date` and `POSIXct` values are converted to the `"YYYY-MM-DD"` strings the library expects, and options that must be arrays (such as `selectedDates`) are boxed for you, so a single date works. JavaScript callbacks can be supplied with `htmlwidgets::JS()`; they run in addition to, not instead of, the Shiny inputs described below. Note that options keep the library's own conventions, so `selectedMonth` counts months from 0 (`11` is December) even though `input$<id>_selected_month` is reported to R as 1-12. |
| `width, height` | Must be a valid CSS unit (like `'100\%'`, `'400px'`, `'auto'`) or a number, which will be coerced to a string and have `'px'` appended. |
| `elementId` | An optional ID for the widget element. |

## Value

An object of class `htmlwidget`.

## Shiny inputs

When rendered in Shiny, the widget reports its state back to the server. For an output with id `"cal"`:

`input$cal_selected` A `Date` vector of the selected dates (length 0 when nothing is selected).

`input$cal_selected_month` Selected month, 1-12.

`input$cal_selected_year` Selected year.

`input$cal_displayed` First day of the displayed month, as a `Date`, updated when the user clicks the arrows.

`input$cal_time` Selected time as a string, e.g. `"14:30"`, when `selectionTimeMode` is enabled.

`input$cal_week` A list with `week` and `year`, when `enableWeekNumbers` is enabled and a week number is clicked.

`input$cal_ready` `TRUE` once the calendar has initialised.

## Input mode

With `options = list(inputMode = TRUE)` the widget renders a text input and shows the calendar as a popup, the behaviour documented under input mode upstream. Pair it with `height = "auto"`.

## Theming

`selectedTheme` accepts `"light"`, `"dark"` or `"system"`. In system mode the library reads the theme from the attribute named by `themeAttrDetect`, which defaults to `"html[data-theme]"`. Shiny apps using **bslib** should set `themeAttrDetect = "html[data-bs-theme]"` to follow the app theme.

## See Also

`VanillaCalendarProxy()` to update a calendar in place from Shiny.

## Examples

```
# Basic calendar with default settings
VanillaCalendar()

# Calendar with custom options
VanillaCalendar(
  options = list(
    type = "default",
    locale = "en",
    selectionDatesMode = "multiple",
    selectedDates = Sys.Date(),
    selectedTheme = "light"
  ),
  width = "500px",
  height = "400px"
)

# Date picker attached to a text input
VanillaCalendar(
```

```
  options = list(inputMode = TRUE, selectionDatesMode = "single"),
  height = "auto"
)
```

---

`VanillaCalendar-shiny` *Shiny bindings for VanillaCalendar*

---

## Description

Output and render functions for using VanillaCalendar within Shiny applications and interactive Rmd documents.

## Usage

```
VanillaCalendarOutput(outputId, width = "100%", height = "400px")

renderVanillaCalendar(expr, env = parent.frame(), quoted = FALSE)
```

## Arguments

| | |
|---|---|
| `outputId` | output variable to read from |
| `width, height` | Must be a valid CSS unit (like `'100%'`, `'400px'`, `'auto'`) or a number, which will be coerced to a string and have `'px'` appended. Use `height = 'auto'` with `inputMode = TRUE`. |
| `expr` | An expression that generates a VanillaCalendar |
| `env` | The environment in which to evaluate `expr`. |
| `quoted` | Is `expr` a quoted expression (with `quote()`)? This is useful if you want to save an expression in a variable. |

## Value

`VanillaCalendarOutput()` returns a Shiny output element; `renderVanillaCalendar()` returns a Shiny render function.

## Examples

```
# Only run examples in interactive R sessions
if (interactive()) {
  library(shiny)
  library(VanillaCalendar)

  ui <- fluidPage(
    VanillaCalendarOutput("calendar"),
    verbatimTextOutput("dates")
  )
```

```
  server <- function(input, output) {
    output$calendar <- renderVanillaCalendar({
      VanillaCalendar(
        options = list(
          type = "default",
          locale = "en",
          selectionDatesMode = "multiple"
        )
      )
    })
    output$dates <- renderPrint(input$calendar_selected)
  }

  shinyApp(ui, server)
}
```

---

`VanillaCalendarProxy` *Update a calendar in place from Shiny*

---

## Description

`VanillaCalendarProxy()` creates a handle to a calendar that is already rendered, so that it can be changed without re-rendering it — the calendar keeps its position, focus and selection instead of flickering back to its initial state. The methods map onto the Calendar instance methods of Vanilla Calendar Pro.

`vcSet()` applies new options to the calendar.

`vcUpdate()` re-renders the calendar with its current options.

`vcShow()` and `vcHide()` show and hide a popup calendar in input mode.

`vcDestroy()` removes the calendar from the page.

## Usage

```
VanillaCalendarProxy(id, session = shiny::getDefaultReactiveDomain())

vcSet(proxy, options, reset = NULL)

vcUpdate(proxy, reset = NULL)

vcShow(proxy)

vcHide(proxy)

vcDestroy(proxy)
```

## Arguments

| | |
|---|---|
| id | The output id of the calendar, as used in VanillaCalendarOutput(). |
| session | The Shiny session object. |
| proxy | A VanillaCalendarProxy object. |
| options | A named list of options to apply, handled exactly as in VanillaCalendar(). |
| reset | Optional named list overriding which parts of the calendar are reset to their option values, with any of the logical entries year, month, dates, time and locale. By default only the parts you are actually setting are reset, so vcSet(proxy, list(selectedTheme = "dark")) leaves the user's selection and the displayed month alone. Pass list(dates = TRUE) to clear a selection you are not replacing. |

## Value

The proxy object, invisibly, so calls can be piped.

## Examples

```
if (interactive()) {
  library(shiny)
  library(VanillaCalendar)

  ui <- fluidPage(
    selectInput("theme", "Theme", c("light", "dark")),
    actionButton("clear", "Clear selection"),
    VanillaCalendarOutput("cal")
  )

  server <- function(input, output, session) {
    output$cal <- renderVanillaCalendar(
      VanillaCalendar(list(selectionDatesMode = "multiple"))
    )

    observeEvent(input$theme, {
      vcSet(VanillaCalendarProxy("cal"), list(selectedTheme = input$theme))
    })

    observeEvent(input$clear, {
      vcSet(VanillaCalendarProxy("cal"), list(selectedDates = list()),
            reset = list(dates = TRUE))
    })
  }

  shinyApp(ui, server)
}
```

# Index