# Package 'ganttify'

June 15, 2026

**Type** Package

**Title** Create Interactive Gantt Charts with Work Breakdown Structure

**Version** 0.2.7

**Description** Create Primavera-style interactive Gantt charts with Work Breakdown Structure (WBS) hierarchy and activities. Features include color-coded WBS items, indented labels, scrollable views for large projects, dynamic date formatting, and the ability to dim past activities. Built on top of 'plotly' for interactive visualizations.

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 3.5.0)

**Imports** plotly (>= 4.9.0), dplyr (>= 1.0.0), htmlwidgets (>= 1.5.0), magrittr, rlang

**RoxygenNote** 7.3.3

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**URL** https://github.com/AhmedAredah/Ganttify

**BugReports** https://github.com/AhmedAredah/Ganttify/issues

**NeedsCompilation** no

**Author** Ahmed Aredah [aut, cre]

**Maintainer** Ahmed Aredah <Ahmed.Aredah@gmail.com>

**Repository** CRAN

**Date/Publication** 2026-06-15 18:30:02 UTC

## Contents

`Ganttify` *Create Interactive Gantt Chart with WBS Structure*

## Description

Creates a Primavera-style interactive Gantt chart with Work Breakdown Structure (WBS) hierarchy and activities. The chart features color-coded WBS items, indented labels, scrollable view for large projects, and dynamic date formatting.

## Usage

```
Ganttify(
  wbs_structure,
  activities,
  chart_title = "Project Gantt Chart with WBS",
  x_range = NULL,
  milestone_lines = NULL,
  color_config = NULL,
  display_config = NULL,
  label_config = NULL,
  bar_config = NULL,
  layout_config = NULL,
  tooltip_config = NULL
)
```

## Arguments

`wbs_structure` A data frame with 3 columns: ID (character), Name (character), and Parent (character). Parent should be "None" or "" for root level items.

`activities` A data frame with 5 required columns, 2 optional columns, and any number of additional columns:

- WBS_ID (character): Associated WBS item identifier
- Activity_ID (character): Unique activity identifier
- Activity_Name (character): Activity name
- Start_Date (character or Date): Planned start date in MM/DD/YYYY format (e.g. "09/15/2024") or Date class
- End_Date (character or Date): Planned end date in MM/DD/YYYY format (e.g. "09/15/2024") or Date class
- Start_Date_Actual (character or Date, optional): Actual start date in MM/DD/YYYY format (e.g. "09/15/2024") or Date class
- End_Date_Actual (character or Date, optional): Actual end date in MM/DD/YYYY format (e.g. "09/15/2024") or Date class. If Start_Date_Actual is provided but End_Date_Actual is missing, the actual bar will show from Start_Date_Actual to today (if today > Start_Date_Actual).

- Additional columns (optional): Any extra columns (e.g., Status, Agency, Priority) are preserved and can be used for attribute-based coloring via color_config with mode="attribute".

When actual dates are provided, activities display as stacked bars: planned on top (solid color) and actual on bottom (diagonal stripe pattern).

`chart_title` Character. Title displayed at the top of the chart. Default "Project Gantt Chart with WBS".

`x_range` Character vector. Date range for x-axis zoom (e.g., c("2024-01-01", "2024-12-31")). If NULL, shows full project range.

`milestone_lines`

Data frame or NULL. Optional milestone markers to display on the chart. Supports both single-date vertical lines and date-range shaded areas. If provided, must be a data frame with the following columns:

- date (required): Either a single date (for vertical line) or a vector of 2 dates (for shaded area). Use a list column to mix both types. Dates can be character in MM/DD/YYYY format (e.g. "09/15/2024") or Date objects.
- label (required): Text label to display on the milestone
- color (optional): Color for line or area (e.g., "red", "#FF0000"). Defaults to color palette.
- dash (optional): Line style for single-date milestones - "solid", "dash", "dot", or "dashdot". Default "dash".
- width (optional): Line width in pixels for single-date milestones. Default 2.
- fill_opacity (optional): Opacity for shaded areas (0-1). Default 0.15. Ignored for lines.
- label_position (optional): Label position - "top", "middle", or "bottom". Default "top".
- label_level (optional): Vertical stacking level for labels - 1 or 2. Level 1 labels are rendered above level 2 labels (further from the chart area). This is useful when multiple milestones are close together and labels would overlap. Default 1.

Example with mixed types:

```
milestones <- data.frame(
  label = c("Kickoff", "Review Period", "Deadline"),
  color = c("green", "blue", "red")
)
milestones$date <- list(
  "01/15/2025",
  c("03/01/2025", "03/31/2025"),
  "12/31/2025"
)
```

Example with label levels (useful for overlapping milestones):

```
milestones <- data.frame(
  label = c("Phase 1 Start", "Phase 2 Start", "Final Review"),
```

```
  color = c("blue", "green", "red"),
  label_level = c(1, 2, 1)  # Level 1 labels appear above level 2
)
milestones$date <- list("2025-01-15", "2025-01-20", "2025-06-30")
```

Default NULL (no milestone markers).

color_config List or NULL. Configuration for chart colors. Structure depends on mode:

- mode="wbs" (default if NULL): Activities inherit colors from parent WBS
  `list(mode = "wbs", wbs = list("W1" = "#FF6B6B", "W2" = "#4ECDC4"))`
- mode="uniform": All activities same color, WBS same color
  `list(mode = "uniform", uniform = list(wbs = "#34495E", activity = "#2ECC71"))`
- mode="attribute": Color activities by attribute column (e.g., Status)

```
list(mode = "attribute",
          attribute = list(column = "Status",
                 mapping = list("completed" = "green", "in-progress" = "orange"),
                         wbs = "#34495E"))
```

If NULL, uses mode="wbs" with default color palette. Default NULL.

display_config List or NULL. Controls visibility of chart elements. Structure:

- wbs: List with show (logical), show_labels (logical), show_names_on_bars (logical)
- activity: List with show (logical), show_names_on_bars (logical)
- milestone: List with hide_label_levels (integer vector or NULL). Suppresses visible text annotations for milestones at the specified label_level values (e.g. `c(1)`, `c(2)`, or `c(1, 2)`). Hover tooltips are unaffected. Default NULL (all labels shown).

Example:

```
list(
 wbs = list(show = TRUE, show_labels = TRUE, show_names_on_bars = TRUE),
  activity = list(show = TRUE, show_names_on_bars = FALSE),
  milestone = list(hide_label_levels = c(1))
)
```

If NULL, uses defaults shown above. Default NULL.

label_config List or NULL. Template strings for labels. Structure:

- activity: List with yaxis (template for y-axis labels) and bar (template for bar labels)
- wbs: List with yaxis and bar templates

Available placeholders for activity: `name`, `id`, `start`, `end`, `start_actual`, `end_actual`, `duration`, `wbs_id` (use with curly braces) Available placeholders for wbs: `name`, `id`, `start`, `end`, `duration` (use with curly braces) Example:

```
list(
 activity = list(yaxis = "{name} ({start} - {end})", bar = "{name}"),
  wbs = list(yaxis = "{name}", bar = "{name}")
)
```

If NULL, uses default template for all labels. Default NULL.

`bar_config` List or NULL. Styling configuration for bars. Structure:

- wbs: List with opacity (0-1) and height (numeric)
- activity: List with opacity (0-1), height (numeric), dim_opacity (0-1), and dim_past_activities (logical)

The dim_past_activities field controls whether activities that end before today are dimmed. When TRUE, completed activities use the dim_opacity value instead of the regular opacity. Note: Short-duration activities are automatically kept visible at any zoom level through dynamic bar width adjustment. The original dates are preserved in hover tooltips. Example:

```
list(
  wbs = list(opacity = 0.3, height = 0.3),
 activity = list(opacity = 1.0, height = 0.8, dim_opacity = 0.3, dim_past_activities = F
)
```

If NULL, uses defaults shown above. Default NULL.

`layout_config` List or NULL. Chart layout settings. Structure:

- buffer_days: Numeric, days to add before/after timeline for margin
- indent_size: Numeric, spaces per indentation level
- max_visible_rows: Numeric, maximum visible rows (enables scrolling)
- y_scroll_position: Numeric or NULL, initial scroll position
- yaxis_label_width: Numeric, width of y-axis label area in pixels (default 300)
- yaxis_label_max_chars: Numeric or NULL, maximum characters for labels before truncating with "..." (NULL = no truncation)
- hover_popup_max_chars: Numeric, maximum characters per line in hover popups before wrapping to next line (default 50)
- show_yaxis_labels: Logical, whether to show y-axis labels (default TRUE). When FALSE, activity labels are hidden. If display_config$wbs$show_labels is TRUE, WBS labels will still be shown; otherwise all y-axis labels are hidden.
- xaxis_position: Character, position of the time (x) axis. One of `"bottom"` (default), `"top"`, or `"both"`. Use `"top"` to keep the axis visible when scrolling down a long chart. Use `"both"` to show axis ticks and labels on both edges simultaneously.

Example:

```
list(
  buffer_days = 30,
  indent_size = 2,
  max_visible_rows = 20,
  y_scroll_position = NULL,
  yaxis_label_width = 300,
  yaxis_label_max_chars = NULL,
  hover_popup_max_chars = 50,
  show_yaxis_labels = TRUE,
```

```
  xaxis_position = "bottom"
)
```

If NULL, uses defaults shown above. Default NULL.

`tooltip_config` List or NULL. Configuration for custom tooltip fields. Structure:

- wbs: Character vector of column names from wbs_structure to display in WBS tooltips. Use a named vector to set a custom display label: `c(col_name = "Display Label")`.
- activity: Character vector of column names from activities to display in activity tooltips. Use a named vector to set a custom display label: `c(col_name = "Display Label")`.
- milestone: Character vector of column names from milestone_lines to display in milestone tooltips. Use a named vector to set a custom display label: `c(col_name = "Display Label")`.

Fields that don't exist in the data or have NA/empty values are automatically hidden. Named and unnamed elements can be mixed freely. Example:

```
list(
 wbs     = c(Owner = "Project Owner", Budget = "Total Budget ($)"),
 activity = c(activity_details = "Activity Details", "Status"),
 milestone = c(Description = "Milestone Description")
)
```

If NULL, only default fields (Start, End, Duration) are shown. Default NULL.

## Value

A plotly object containing the interactive Gantt chart. Can be displayed directly or saved using htmlwidgets::saveWidget().

## Examples

```
# Load test data
data(test_project)

# Basic Gantt chart with WBS colors
chart <- Ganttify(
  wbs_structure = test_project$wbs_structure,
  activities = test_project$activities,
  color_config = list(mode = "wbs", wbs = test_project$colors)
)
chart

# Uniform color mode
chart <- Ganttify(
  wbs_structure = test_project$wbs_structure,
  activities = test_project$activities,
  color_config = list(
    mode = "uniform",
    uniform = list(wbs = "#34495E", activity = "#2ECC71")
  )
```

```
)
chart

# Attribute-based coloring (requires extra column in activities)
# Add a Status column to activities dataframe
activities_with_status <- test_project$activities
activities_with_status$Status <- sample(c("completed", "in-progress", "pending"),
                                        nrow(activities_with_status), replace = TRUE)
chart <- Ganttify(
  wbs_structure = test_project$wbs_structure,
  activities = activities_with_status,
  color_config = list(
    mode = "attribute",
    attribute = list(
      column = "Status",
     mapping = list("completed" = "green", "in-progress" = "orange", "pending" = "gray"),
      wbs = "#34495E"
    )
  )
)
chart

# WBS-only view using display_config
chart <- Ganttify(
  wbs_structure = test_project$wbs_structure,
  activities = test_project$activities,
  display_config = list(activity = list(show = FALSE))
)
chart

# Custom labels showing date ranges
chart <- Ganttify(
  wbs_structure = test_project$wbs_structure,
  activities = test_project$activities,
  label_config = list(
    activity = list(yaxis = "{name} ({start} - {end})")
  )
)
chart

# Customize bar heights and enable dimming for past activities
chart <- Ganttify(
  wbs_structure = test_project$wbs_structure,
  activities = test_project$activities,
  bar_config = list(
    wbs = list(opacity = 0.5, height = 0.4),
    activity = list(height = 0.9, dim_past_activities = TRUE, dim_opacity = 0.4)
  )
)
chart

# Add "today" line as a milestone
chart <- Ganttify(
```

```
  wbs_structure = test_project$wbs_structure,
  activities = test_project$activities,
  milestone_lines = data.frame(
    date = Sys.Date(),
    label = "Today",
    color = "red"
  )
)
chart

# Narrow label area with truncation
chart <- Ganttify(
  wbs_structure = test_project$wbs_structure,
  activities = test_project$activities,
  layout_config = list(
    yaxis_label_width = 200,
    yaxis_label_max_chars = 25
  )
)
chart

# Custom tooltip fields (add extra columns to show in hover popups)
# Add custom columns to activities, WBS, and milestone data
activities_extended <- test_project$activities
activities_extended$Status <- sample(c("On Track", "Delayed", "Complete"),
                                     nrow(activities_extended), replace = TRUE)
activities_extended$Agency <- "TTI"

wbs_extended <- test_project$wbs_structure
wbs_extended$Owner <- "Project Manager"

milestones <- data.frame(
  label = c("Kickoff", "Deadline"),
  color = c("green", "red"),
  Description = c("Project start", "Final delivery"),
  stringsAsFactors = FALSE
)
milestones$date <- list("01/15/2025", "12/31/2025")

chart <- Ganttify(
  wbs_structure = wbs_extended,
  activities = activities_extended,
  milestone_lines = milestones,
  tooltip_config = list(
    wbs = c("Owner"),
    activity = c("Status", "Agency"),
    milestone = c(Description = "Notes")
  )
)
chart
```

`test_project` *Test Project Data for Ganttify*

## Description

A sample project dataset for testing and demonstrating the Ganttify package. Contains a complete Work Breakdown Structure (WBS), activities, and custom colors for an example software development project.

## Usage

```
test_project
```

## Format

A list with 3 components:

**wbs_structure** A data frame with 8 rows and 3 columns:

- ID: WBS item identifier (W1-W8)
- Name: WBS item name (e.g., "Project Summary", "Design Phase")
- Parent: Parent WBS ID or "None" for root items

**activities** A data frame with 15 rows and 7 columns:

- WBS_ID: Associated WBS item identifier
- Activity_ID: Activity identifier (A1-A15)
- Activity_Name: Activity name (e.g., "Design UI", "Code Frontend")
- Start_Date: Planned start date (YYYY-MM-DD or MM/DD/YYYY character or Date class)
- End_Date: Planned end date (YYYY-MM-DD or MM/DD/YYYY character or Date class)
- Start_Date_Actual: Actual start date (YYYY-MM-DD or MM/DD/YYYY character or Date class, some NA for not started)
- End_Date_Actual: Actual end date (YYYY-MM-DD or MM/DD/YYYY character or Date class, some NA for in-progress)

Includes examples of on-time, delayed, ahead-of-schedule, and in-progress activities.

**colors** A named list of 8 colors:

- Each WBS item (W1-W8) is assigned a custom hex color

## Source

Example software development project

## Examples

```
# Load the test data
data(test_project)

# View structure
str(test_project)

# Create a Gantt chart

chart <- Ganttify(
  wbs_structure = test_project$wbs_structure,
  activities = test_project$activities,
  color_config = list(mode = "wbs", wbs = test_project$colors)
)
chart
```

# Index