Guide: Transition Engine

1 Introduction

The `render-transition` function is designed to automatically generate “roadmap” or “summary” slides when the document structure changes (e.g., entering a new section).

Previously, it required many manual parameters. Now, it is designed to be **configured globally** via `navigator-config`, allowing a clean one-line integration in your document.

2 Global Configuration

Instead of passing parameters to every `render-transition` call, you can set them once at the beginning of your document:

```
#import "@preview/navigator:0.2.0": navigator-config, render-transition

#navigator-config.update(c => {
  c.mapping = (section: 1, subsection: 2)
  c.slide-func = my-presentation-engine.slide
  c.theme-colors = (primary: navy, accent: orange)
  c
})

// Now you can use a simple one-liner:
#show heading: render-transition
```

3 Function documentation

`render-transition(h, transitions: (:), mapping: (:), ...)`

3.1 Parameters Reference

Most parameters default to `auto`, which means they will be resolved from the global `navigator-config` state.

Option	Type	Description
`h`	heading	**Mandatory**. The heading object intercepted by the show rule.
`slide-func`	function | auto	A callback `(fill: color, body: content) => content` used to create the slide. If `auto`, resolved from global config.
`transitions`	dict	Detailed configuration for the transition engine. These settings are merged with the global config defaults.
`mapping`	dict | auto	Maps heading levels to roles (e.g., `(section: 1, subsection: 2)`). If `auto`, resolved from global config.
`theme-colors`	dict | auto	Dictionary containing `primary` and `accent` colors. If `auto`, resolved from global config.
`show-heading-numbering`	bool | auto	Whether to display heading numbers in the roadmap. If `auto`, resolved from global config.
`numbering-format`	string | auto	Numbering format (e.g., `"1.1"`). If `auto`, resolved from global config.
`base-text-size`	length | auto	Base font size for the roadmap text. Default: `auto`.
`base-text-font`	string | auto	Font family for the roadmap text. Default: `auto`.

Option	Type	Description
`top-padding`	relative | length	Vertical spacing added above the roadmap in **Standard Mode**. Default: `40%`.
`content-padding`	length | dict	Padding around the roadmap content in **Standard Mode**. Default: `(x: 10%)`.
`content-align`	alignment	Alignment of the roadmap content in **Standard Mode**. Default: `top + left`.
`content-wrapper`	function	A callback `(roadmap, heading, active-state) => content` to completely override the slide layout (**Expert Mode**).
`max-length`	int | dict | auto	Maximum length of titles before truncation. If `auto`, resolved from global config.
`use-short-title`	bool | dict | auto	Whether to use short titles. If `auto`, resolved from global config.

3.2 The `transitions` dictionary

This parameter allows fine-tuning the behavior and appearance of transition slides.

Key	Type	Description
`enabled`	bool	Global switch for transitions. Default: `true`.
`max-level`	int	The maximum heading level that triggers a transition. Default: `3`.
`background`	color | string	Background type: `"theme"` (uses `primary` color), `"none"`, or an explicit color.
`filter`	function	A callback `(heading) => bool` to programmatically enable/disable transitions for specific headings.
`style`	dict	Controls typography: `inactive-opacity` (default `0.3`), `completed-opacity` (default `0.6`), `active-weight` (default `"bold"`).
`sections`	dict	Override for section-level transitions. Contains `enabled`, `visibility` (dict), and `background`.
`subsections`	dict	Override for subsection-level transitions. Contains `enabled`, `visibility` (dict), and `background`.

3.2.1 Visibility Logic

For each transition role (`parts`, `sections`, `subsections`), you can define which hierarchy levels are visible using the `visibility` key. The behavior depends on the active heading:

- `"all"`: Show all headings at this level throughout the document.
- `"current"`: Only show the heading that is currently active (or the parent of the active one).
- `"current-parent"`: Show all siblings of the active heading (i.e., all headings at this level that share the same parent).
- `"none"`: Completely hide this level from the roadmap.

3.3 Customizing the Layout

You have two ways to control the roadmap appearance on the slide:

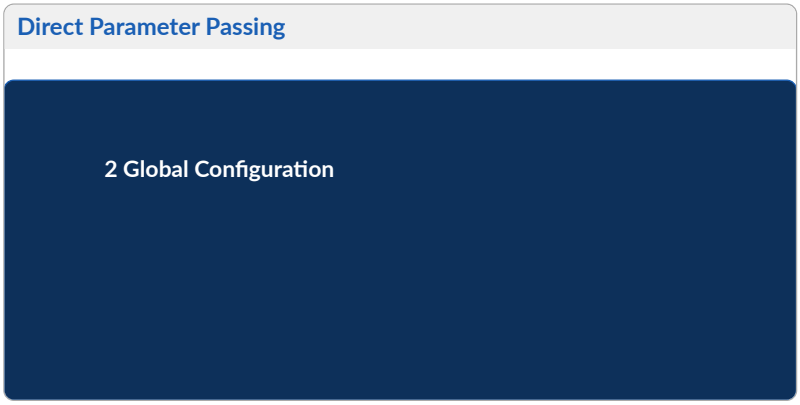
1. **Standard Mode**: Use `top-padding`, `content-padding`, and `content-align` to position the roadmap. This is quick and works for most cases.
2. **Expert Mode**: Use `content-wrapper` to take full control. Note that if `content-wrapper` is provided, the standard padding and alignment parameters are ignored.

4 Basic usage

4.1 Explicit mode (Legacy/Direct)

You can still pass all parameters manually if you prefer total isolation for specific headings.

```
render-transition(
  h,
  mapping: (section: 1, subsection: 2),
  theme-colors: (primary: navy),
  slide-func: my-slide-func,
)
```



4.2 Minimalist mode (Recommended)

By using `navigator-config.update(...)`, your show rule becomes extremely clean:

```
#show heading: render-transition
```

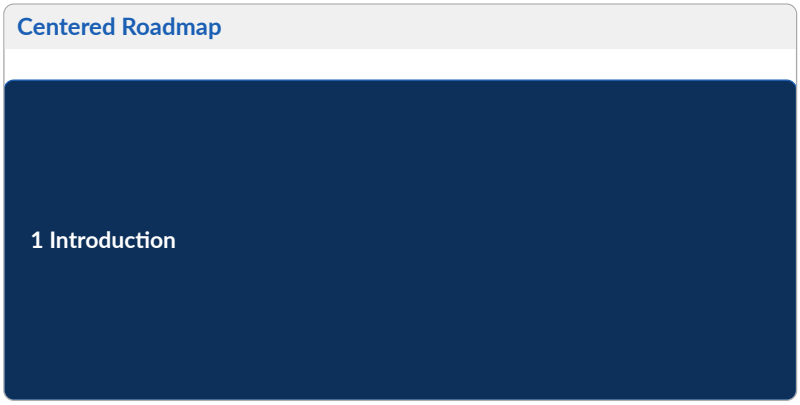
This is the recommended way to use Navigator in modern presentations.

5 Layout & Positioning Control

5.1 Padding and Alignment

Use `content-padding` and `content-align` to place the roadmap exactly where you want it.

```
render-transition(
  h,
  top-padding: 0pt,
  content-align: center + horizon,
  content-padding: 0pt,
  slide-func: my-slide-func,
)
```



5.2 The Wrapper Mode (Total Control)

The `content-wrapper` parameter offers complete freedom over the slide layout. It allows you to place the roadmap alongside other elements (images, titles, decorations) or to wrap it in complex layouts.

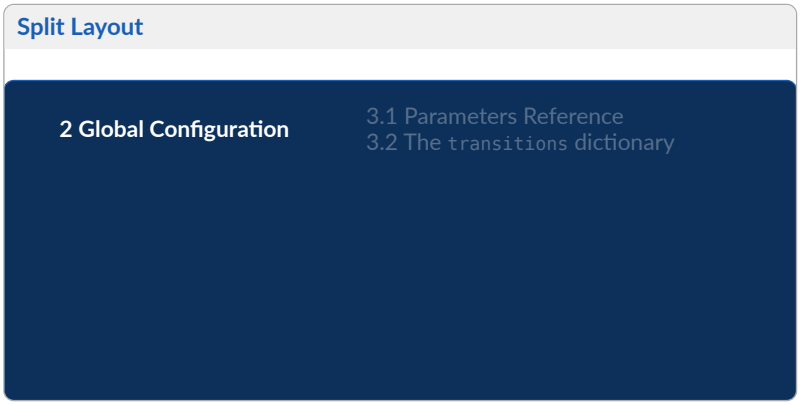
The callback function receives three arguments:

1. `roadmap`: The progressive outline component generated by Navigator.
2. `heading`: The specific heading object that triggered this transition.
3. `active-state`: A dictionary `{ h1, h2, h3 }` containing the active hierarchy (useful to retrieve the parent Part title when entering a Section).

5.2.1 Example 1: Split Layout (Title + Roadmap)

In this example, we place the current section title on the left and the roadmap on the right. We also disable the section display in the roadmap to avoid repetition.

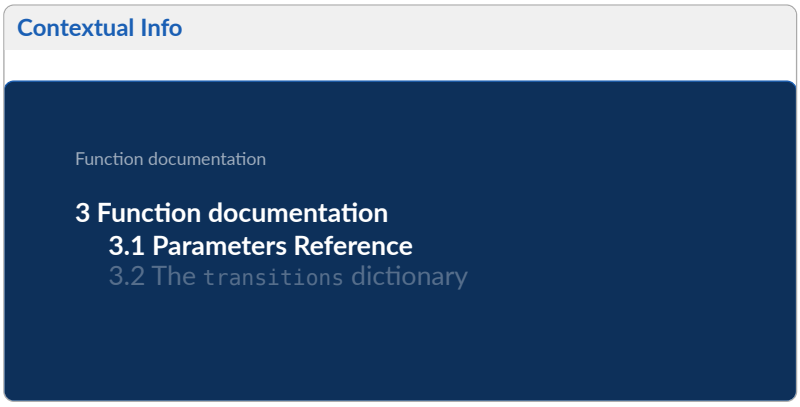
```
content-wrapper: (roadmap, h, active) => {
  grid(
    columns: (1fr, 1.5fr),
    align(center + horizon,
      text(size: 1.5em, weight: "bold",
format-heading(h))
    ),
    align(left + horizon, roadmap)
  )
},
transitions: (
  sections: (visibility: (section: "none",
subsection: "all"))
)
```



5.2.2 Example 2: Contextual Header

Use `active-state` to display the parent heading (e.g., the Part) above the roadmap.

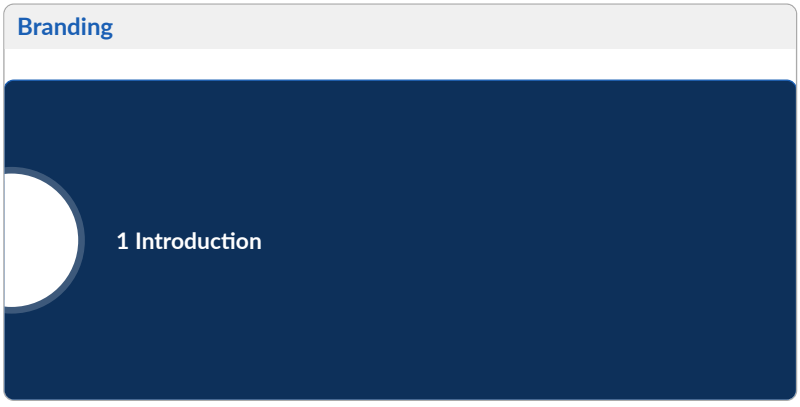
```
content-wrapper: (roadmap, h, active) => {
  set align(left + top)
  if active.h1 != none {
    text(size: 0.8em, fill:
white.transparentize(40%),
smallcaps(active.h1.body))
    v(0.5em)
  }
  roadmap
}
```



5.2.3 Example 3: Branding (Logo + Roadmap)

You can easily integrate graphical elements.

```
content-wrapper: (roadmap, h, active) => {
  stack(dir: ltr, spacing: 2em,
    align(horizon, image('logo.png')),
    align(horizon, roadmap)
  )
}
```

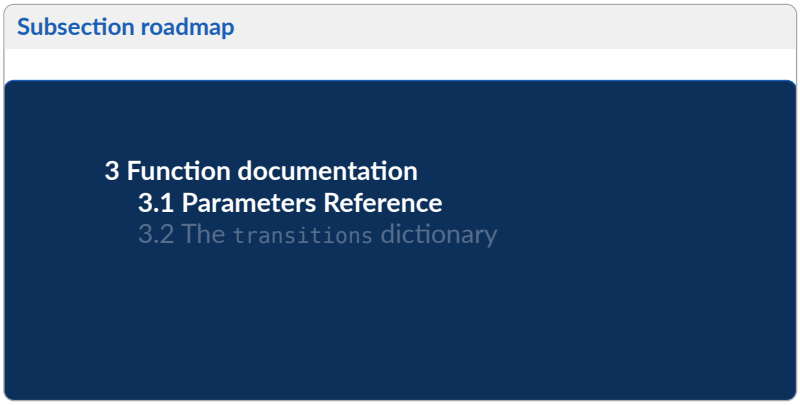


6 Advanced Customization

6.1 Selective visibility

You can configure different roadmap layouts for sections and subsections.

```
transitions: (
  subsections: (
    visibility: (
      section: 'current',
      subsection: 'current-parent'
    )
  )
)
```



7 Short Titles & Truncation

`render-transition` inherits the short title and truncation capabilities of `progressive-outline`. This is especially useful for roadmap slides where titles can be very long.

```
render-transition(
  h,
  use-short-title: true,
  max-length: 15,
  slide-func: my-slide-func,
)
```