

# Guide: Progressive Outline

## 1 Introduction

The progressive-outline function is a versatile component for building roadmaps, sidebars, and breadcrumbs. It displays a hierarchical view of the document structure that evolves as the reader progresses.

Previously, it required manual styling for every call. Now, it is designed to be **configured globally** via navigator-config, ensuring consistency across your document.

## 2 Global Configuration

You can define the default appearance and behavior of all outlines at the beginning of your document:

```
#import "@preview/navigator:0.2.0": navigator-config, progressive-outline
```

```
#navigator-config.update(c => {  
  c.progressive-outline = (  
    level-1-mode: "all",  
    level-2-mode: "current-parent",  
    text-styles: (  
      level-1: (active: (weight: "bold", fill: navy)),  
      level-2: (active: (weight: "regular", fill: gray))  
    ),  
    spacing: (indent-2: 2em)  
  )  
  c  
})
```

```
// Use it with zero arguments to respect global config:  
#progressive-outline()
```

## 3 Function documentation

This section details all the parameters available for the progressive-outline function. Most parameters default to auto, which means they will be resolved from the global navigator-config state.

Option	Type	Effect & Expected Values
level-X-mode	string   auto	Defines the visibility of level X (1, 2, or 3). Values: "all", "current", "current-parent", "none". Defaults to global config.
layout	string	Switch between "vertical" (default) and "horizontal" rendering.
separator	content   str	Separator displayed between items in horizontal layout. Ignored in vertical mode.
text-styles	dict   auto	Styles passed to #text. Merged with global config. You can use a float (e.g., 0.5) as a shortcut for opacity inheritance.
spacing	dict   auto	Controls spacing (v-between-X-Y, indent-X, h-spacing). Merged with global config.
show-numbering	bool   auto	Whether to display heading numbering. Defaults to global config.
numbering-format	str   func   auto	Typst numbering format or custom function. If auto, resolved from global config or heading settings.

Option	Type	Effect & Expected Values
match-page-only	bool	If true, considers a heading active if it is on the same page (useful for sidebars). Default: false.
filter	func	A callback function (heading) => bool to programmatically include or exclude headings.
marker	content   dict   func	Content displayed before the item. Can be static, a dict by state, or a function (state, level) => content.
clickable	bool	Enables clickable links on headings. Default: true.
max-length	int   dict   auto	Maximum length of titles before truncation. Defaults to global config.
use-short-title	bool   dict   auto	Whether to use manual short titles. Defaults to global config.

## 4 Layout Modes

progressive-outline supports two main layout modes: "vertical" (the default, based on a grid) and "horizontal" (based on a stack).

### 4.1 Vertical Layout

This is the default mode, optimized for sidebars and roadmap slides. It supports complex indentation and vertical spacing between different levels.

### 4.2 Horizontal Layout

The horizontal mode is ideal for headers, footers, and breadcrumbs. Elements are placed side-by-side and can be separated by custom content.

```
progressive-outline(
  layout: 'horizontal',
  level-1-mode: 'current',
  level-2-mode: 'current',
  separator: ' > ',
  show-numbering: true
)
```

#### Horizontal Breadcrumb

**4 Layout Modes** > 4.2 Horizontal Layout

```
progressive-outline(
  layout: 'horizontal',
  level-1-mode: 'all',
  level-2-mode: 'none',
  separator: [ | ],
  spacing: (h-spacing: 1em)
)
```

#### Horizontal Navigation Bar

**1 Introduction** | **2 Global Configuration** | **3 Function documentation**

## 5 Navigation & Interactivity

By default, the outline is interactive: clicking on a section title navigates directly to the corresponding slide in the PDF.

```
progressive-outline()
```

### Non-clickable Outline

- 1 Introduction
- 2 Global Configuration
- 3 Function documentation
- 4 Layout Modes
- 5 Navigation & Interactivity**
- 6 Visibility
- 7 Style Customization
- 8 Customizable Markers
- 9 The anti-jitter mechanism
- 10 Fine-grained spacing management
- 11 Numbering system
- 12 Filtering Content
- 13 Advanced Behavior
- 14 Title Truncation
- 15 Short Titles
- 16 Additional information

In some print-focused or strict layout scenarios, you might want to disable this interactivity.

```
progressive-outline(  
  clickable: false  
)
```

### Non-clickable Outline

- 1 Introduction
- 2 Global Configuration
- 3 Function documentation
- 4 Layout Modes
- 5 Navigation & Interactivity**
- 6 Visibility
- 7 Style Customization
- 8 Customizable Markers
- 9 The anti-jitter mechanism
- 10 Fine-grained spacing management
- 11 Numbering system
- 12 Filtering Content
- 13 Advanced Behavior
- 14 Title Truncation
- 15 Short Titles
- 16 Additional information

## 6 Visibility

This section covers the level-X-mode parameters.

## 6.1 The ‘current-parent’ mode

The current-parent mode is the most powerful: it only displays the “siblings” of the current element. This allows you to see the plan of the current section without being distracted by other chapters.

```
progressive-outline(  
  level-1-mode: 'all',  
  level-2-mode: 'current-parent'  
)
```

### Visibility Demonstration H2

- 1 Introduction
- 2 Global Configuration
- 3 Function documentation
- 4 Layout Modes
- 5 Navigation & Interactivity
- 6 Visibility
  - 6.1 The ‘current-parent’ mode
  - 6.2 Isolation via ‘current’ mode
  - 6.3 Deep nesting (Level 3)
- 7 Style Customization
- 8 Customizable Markers
- 9 The anti-jitter mechanism
- 10 Fine-grained spacing management
- 11 Numbering system
- 12 Filtering Content
- 13 Advanced Behavior
- 14 Title Truncation
- 15 Short Titles
- 16 Additional information

## 6.2 Isolation via ‘current’ mode

If you want an ultra-minimalist rendering, the current mode hides everything except the exact entry where you are located.

```
progressive-outline(  
  level-1-mode: 'current',  
  level-2-mode: 'none'  
)
```

### Isolated Visibility Demonstration

#### 6 Visibility

## 6.3 Deep nesting (Level 3)

For complex structures, you can enable Level 3. Using current-parent will show siblings at the current depth.

### 6.3.1 Deep Component A

### 6.3.2 Deep navigation test

```
progressive-outline(  
  level-2-mode: 'all',  
  level-3-mode: 'current-parent'  
)
```

#### Level 3 Siblings

##### 1 Introduction

##### 2 Global Configuration

##### 3 Function documentation

##### 4 Layout Modes

###### 4.1 Vertical Layout

###### 4.2 Horizontal Layout

##### 5 Navigation & Interactivity

##### 6 Visibility

###### 6.1 The 'current-parent' mode

###### 6.2 Isolation via 'current' mode

###### 6.3 Deep nesting (Level 3)

###### 6.3.1 Deep Component A

###### 6.3.2 Deep navigation test

##### 7 Style Customization

###### 7.1 The 3-state system

###### 7.2 Advanced Opacity & Inheritance

##### 8 Customizable Markers

###### 8.1 Static Marker

###### 8.2 State-based Markers (Dictionary)

###### 8.3 Dynamic Markers (Function)

###### 8.4 Marker Alignment

##### 9 The anti-jitter mechanism

###### 9.1 Colors and decorations

##### 10 Fine-grained spacing management

###### 10.1 Inter-level spacing

###### 10.2 Horizontal indentation

##### 11 Numbering system

###### 11.1 Complex hierarchical formats

###### 11.2 Advanced textual prefixes

##### 12 Filtering Content

###### 12.1 Label-based filtering

###### 12.2 Logic-based filtering

###### 12.3 Recursive filtering

##### 13 Advanced Behavior

###### 13.1 Page-based matching

##### 14 Title Truncation

###### 14.1 Global Truncation

###### 14.2 Per-level Truncation

###### 14.3 Horizontal Truncation

##### 15 Short Titles

###### 15.1 Basic Usage

###### 15.2 Disabling Short Titles

##### 16 Additional information

## 7 Style Customization

The function allows you to modify the appearance of headings based on their state (**completed**, **active**, or **inactive**).

## 7.1 The 3-state system

By default, headings can be in one of three states:

- **completed:** The heading has already been passed.
- **active:** This is the current heading.
- **inactive:** The heading is yet to come.

```
text-styles: (  
  level-1: (  
    active: (fill: eastern, weight:  
'bold'),  
    completed: (fill: gray.lighten(50%)),  
    inactive: (fill: black)  
  )  
)
```

### Past, Present, Future

1 Introduction

2 Global Configuration

3 Function documentation

4 Layout Modes

5 Navigation & Interactivity

6 Visibility

7 Style Customization

8 Customizable Markers

9 The anti-jitter mechanism

10 Fine-grained spacing management

11 Numbering system

12 Filtering Content

13 Advanced Behavior

14 Title Truncation

15 Short Titles

16 Additional information

## 7.2 Advanced Opacity & Inheritance

Instead of redefining the full style for inactive or completed states, you can use smart inheritance to adapt the active style.

### 7.2.1 The Float Shortcut (Clone & Fade)

Pass a number (0.0 to 1.0) to automatically clone the active style and apply transparency. 0.2 means 20% opacity (very faint), 1.0 means fully opaque.

```

text-styles: (
  level-1: (
    active: (fill: red, weight: 'black'),
    inactive: 0.2,    // Future: very faint
(20%)
    completed: 0.5    // Past: semi-
transparent (50%)
  )
)

```

## Auto-Fade Shortcut

- 1 Introduction**
- 2 Global Configuration**
- 3 Function documentation**
- 4 Layout Modes**
- 5 Navigation & Interactivity**
- 6 Visibility**
- 7 Style Customization**
- 8 Customizable Markers
- 9 The anti-jitter mechanism
- 10 Fine-grained spacing management
- 11 Numbering system
- 12 Filtering Content
- 13 Advanced Behavior
- 14 Title Truncation
- 15 Short Titles
- 16 Additional information

### 7.2.2 Partial Inheritance (Mix & Match)

You can also use a dictionary with an opacity key. This allows you to inherit the active color (faded) while overriding other properties (like weight).

```

text-styles: (
  level-1: (
    active: (fill: blue, weight: 'black'),
    inactive: (
      opacity: 0.5,    // 50% of active
color
      weight: 'regular' // But force
regular weight
    )
  )
)

```

## Fade + Weight Change

- 1 Introduction**
- 2 Global Configuration**
- 3 Function documentation**
- 4 Layout Modes**
- 5 Navigation & Interactivity**
- 6 Visibility**
- 7 Style Customization**
- 8 Customizable Markers
- 9 The anti-jitter mechanism
- 10 Fine-grained spacing management
- 11 Numbering system
- 12 Filtering Content
- 13 Advanced Behavior
- 14 Title Truncation
- 15 Short Titles
- 16 Additional information

## 8 Customizable Markers

You can add visual indicators (icons, arrows, etc.) before each item using the marker parameter.

## 8.1 Static Marker

The simplest usage is to pass a single content element (like a symbol) that will be used for all items.

```
progressive-outline(  
  marker: sym.triangle.filled.small  
)
```

### Static Symbol

- ▲ 1 Introduction
- ▲ 2 Global Configuration
- ▲ 3 Function documentation
- ▲ 4 Layout Modes
- ▲ 5 Navigation & Interactivity
- ▲ 6 Visibility
- ▲ 7 Style Customization
- ▲ 8 Customizable Markers
- ▲ 9 The anti-jitter mechanism
- ▲ 10 Fine-grained spacing management
- ▲ 11 Numbering system
- ▲ 12 Filtering Content
- ▲ 13 Advanced Behavior
- ▲ 14 Title Truncation
- ▲ 15 Short Titles
- ▲ 16 Additional information

## 8.2 State-based Markers (Dictionary)

You can define different markers for active, inactive, and completed states using a dictionary.

```
progressive-outline(  
  marker: (  
    active: sym.arrow.r,  
    completed: sym.checkmark,  
    inactive: sym.circle.small  
  )  
)
```

### State Indicators

- ✓ 1 Introduction
- ✓ 2 Global Configuration
- ✓ 3 Function documentation
- ✓ 4 Layout Modes
- ✓ 5 Navigation & Interactivity
- ✓ 6 Visibility
- ✓ 7 Style Customization
- 8 Customizable Markers
  - 9 The anti-jitter mechanism
  - 10 Fine-grained spacing management
  - 11 Numbering system
  - 12 Filtering Content
  - 13 Advanced Behavior
  - 14 Title Truncation
  - 15 Short Titles
  - 16 Additional information



## 8.3 Dynamic Markers (Function)

For total control, pass a function (state, level) => content. This allows you to vary markers based on depth level and status.

```
progressive-outline(  
  marker: (state, level) => {  
    if level == 1 { sym.star.filled }  
    else if state ==  
  'active' { sym.arrow.r }  
    else { sym.circle.filled.tiny }  
  }  
)
```

### Advanced Logic

- ★ 1 Introduction
- ★ 2 Global Configuration
- ★ 3 Function documentation
- ★ 4 Layout Modes
  - 4.1 Vertical Layout
  - 4.2 Horizontal Layout
- ★ 5 Navigation & Interactivity
- ★ 6 Visibility
  - 6.1 The 'current-parent' mode
  - 6.2 Isolation via 'current' mode
  - 6.3 Deep nesting (Level 3)
- ★ 7 Style Customization
  - 7.1 The 3-state system
  - 7.2 Advanced Opacity & Inheritance
- ★ 8 Customizable Markers
  - 8.1 Static Marker
  - 8.2 State-based Markers (Dictionary)
  - 8.3 Dynamic Markers (Function)
  - 8.4 Marker Alignment
- ★ 9 The anti-jitter mechanism
  - 9.1 Colors and decorations
- ★ 10 Fine-grained spacing management
  - 10.1 Inter-level spacing
  - 10.2 Horizontal indentation
- ★ 11 Numbering system
  - 11.1 Complex hierarchical formats
  - 11.2 Advanced textual prefixes
- ★ 12 Filtering Content
  - 12.1 Label-based filtering
  - 12.2 Logic-based filtering
  - 12.3 Recursive filtering
- ★ 13 Advanced Behavior
  - 13.1 Page-based matching
- ★ 14 Title Truncation
  - 14.1 Global Truncation
  - 14.2 Per-level Truncation
  - 14.3 Horizontal Truncation
- ★ 15 Short Titles
  - 15.1 Basic Usage
  - 15.2 Disabling Short Titles
- ★ 16 Additional information

## 8.4 Marker Alignment

Use the spacing parameter to fine-tune layout:

- marker-gap: Space between marker and text (default 0.5em).
- marker-width: Fixed width for the marker container (useful for alignment).

```
progressive-outline(  
  marker: (active: sym.arrow.r),  
  spacing: (  
    marker-gap: 1em,  
    marker-width: 1.5em  
  )  
)
```

## Aligned Markers

- 1 Introduction
- 2 Global Configuration
- 3 Function documentation
- 4 Layout Modes
- 5 Navigation & Interactivity
- 6 Visibility
- 7 Style Customization
- 8 Customizable Markers
- 9 The anti-jitter mechanism
- 10 Fine-grained spacing management
- 11 Numbering system
- 12 Filtering Content
- 13 Advanced Behavior
- 14 Title Truncation
- 15 Short Titles
- 16 Additional information

## 9 The anti-jitter mechanism

Anti-jitter ensures that switching from a thin font to a bold one doesn't move the text. We use a ghost box to reserve the maximum space required.

```
text-styles: (  
  level-1: (  
    active: (weight: 'black', fill:  
eastern, size: 1.2em),  
    inactive: (weight: 'light', fill:  
gray, size: 1.2em)  
  )  
)
```

## Stability Test H1

- 1 Introduction
- 2 Global Configuration
- 3 Function documentation
- 4 Layout Modes
- 5 Navigation & Interactivity
- 6 Visibility
- 7 Style Customization
- 8 Customizable Markers
- 9 The anti-jitter mechanism**
- 10 Fine-grained spacing management
- 11 Numbering system
- 12 Filtering Content
- 13 Advanced Behavior
- 14 Title Truncation
- 15 Short Titles
- 16 Additional information

## 9.1 Colors and decorations

Each level can have its own rules for colors, italics, or bold.

```
text-styles: (  
  level-2: (  
    active: (style: 'italic', fill: blue,  
weight: 'bold'),  
    inactive: (fill: luma(200))  
  )  
)
```

### Creative Style H2

#### 1 Introduction

#### 2 Global Configuration

#### 3 Function documentation

#### 4 Layout Modes

- 4.1 Vertical Layout
- 4.2 Horizontal Layout

#### 5 Navigation & Interactivity

#### 6 Visibility

- 6.1 The 'current-parent' mode
- 6.2 Isolation via 'current' mode
- 6.3 Deep nesting (Level 3)

#### 7 Style Customization

- 7.1 The 3-state system
- 7.2 Advanced Opacity & Inheritance

#### 8 Customizable Markers

- 8.1 Static Marker
- 8.2 State-based Markers (Dictionary)
- 8.3 Dynamic Markers (Function)
- 8.4 Marker Alignment

#### 9 The anti-jitter mechanism

##### *9.1 Colors and decorations*

#### 10 Fine-grained spacing management

- 10.1 Inter-level spacing
- 10.2 Horizontal indentation

#### 11 Numbering system

- 11.1 Complex hierarchical formats
- 11.2 Advanced textual prefixes

#### 12 Filtering Content

- 12.1 Label-based filtering
- 12.2 Logic-based filtering
- 12.3 Recursive filtering

#### 13 Advanced Behavior

- 13.1 Page-based matching

#### 14 Title Truncation

- 14.1 Global Truncation
- 14.2 Per-level Truncation
- 14.3 Horizontal Truncation

#### 15 Short Titles

- 15.1 Basic Usage
- 15.2 Disabling Short Titles

#### 16 Additional information

## 10 Fine-grained spacing management

The spacing dictionary sculpts the rhythm.

## **10.1 Inter-level spacing**

You can define the exact space between an H1 heading and an H2 heading, or between two headings of the same level.

```
spacing: (  
  v-between-1-1: 2em,  
  v-between-1-2: 1.2em,  
  v-between-2-2: 0.8em,  
  v-between-2-1: 1.5em  
)
```

## Airy Vertical Rhythm

### 1 Introduction

### 2 Global Configuration

### 3 Function documentation

### 4 Layout Modes

#### 4.1 Vertical Layout

#### 4.2 Horizontal Layout

### 5 Navigation & Interactivity

### 6 Visibility

#### 6.1 The 'current-parent' mode

#### 6.2 Isolation via 'current' mode

#### 6.3 Deep nesting (Level 3)

### 7 Style Customization

#### 7.1 The 3-state system

#### 7.2 Advanced Opacity & Inheritance

### 8 Customizable Markers

#### 8.1 Static Marker

#### 8.2 State-based Markers (Dictionary)

#### 8.3 Dynamic Markers (Function)

#### 8.4 Marker Alignment

### 9 The anti-jitter mechanism

#### 9.1 Colors and decorations

### 10 Fine-grained spacing management

#### 10.1 Inter-level spacing

#### 10.2 Horizontal indentation

### 11 Numbering system

#### 11.1 Complex hierarchical formats

#### 11.2 Advanced textual prefixes

### 12 Filtering Content

#### 12.1 Label-based filtering

#### 12.2 Logic-based filtering

#### 12.3 Recursive filtering

### 13 Advanced Behavior

#### 13.1 Page-based matching

### 14 Title Truncation

#### 14.1 Global Truncation

#### 14.2 Per-level Truncation

#### 14.3 Horizontal Truncation

### 15 Short Titles

### 16 Additional Information

## 10.2 Horizontal indentation

Indentation defines the offset to the right for each depth level.

```
spacing: (  
  indent-2: 3em,  
  indent-3: 6em  
)
```

### Marked Indentation

#### 1 Introduction

#### 2 Global Configuration

#### 3 Function documentation

#### 4 Layout Modes

##### 4.1 Vertical Layout

##### 4.2 Horizontal Layout

#### 5 Navigation & Interactivity

#### 6 Visibility

##### 6.1 The 'current-parent' mode

##### 6.2 Isolation via 'current' mode

##### 6.3 Deep nesting (Level 3)

###### 6.3.1 Deep Component A

###### 6.3.2 Deep navigation test

#### 7 Style Customization

##### 7.1 The 3-state system

##### 7.2 Advanced Opacity & Inheritance

###### 7.2.1 The Float Shortcut (Clone & Fade)

###### 7.2.2 Partial Inheritance (Mix & Match)

#### 8 Customizable Markers

##### 8.1 Static Marker

##### 8.2 State-based Markers (Dictionary)

##### 8.3 Dynamic Markers (Function)

##### 8.4 Marker Alignment

#### 9 The anti-jitter mechanism

##### 9.1 Colors and decorations

#### 10 Fine-grained spacing management

##### 10.1 Inter-level spacing

##### 10.2 Horizontal indentation

#### 11 Numbering system

##### 11.1 Complex hierarchical formats

##### 11.2 Advanced textual prefixes

#### 12 Filtering Content

##### 12.1 Label-based filtering

##### 12.2 Logic-based filtering

##### 12.3 Recursive filtering

#### 13 Advanced Behavior

##### 13.1 Page-based matching

#### 14 Title Truncation

##### 14.1 Global Truncation

##### 14.2 Per-level Truncation

##### 14.3 Horizontal Truncation

#### 15 Short Titles

##### 15.1 Basic Usage

##### 15.2 Disabling Short Titles

#### 16 Additional information

## 11 Numbering system

The function relies on Typst's native engine.

### 11.1 Complex hierarchical formats

The `numbering-format` parameter accepts all standard Typst models (1, a, i, I, A).

```
show-numbering: true,  
numbering-format: 'I.a.1. '
```

## Legal Format

### I. Introduction

### II. Global Configuration

### III. Function documentation

### IV. Layout Modes

IV.a. Vertical Layout

IV.b. Horizontal Layout

### V. Navigation & Interactivity

### VI. Visibility

VI.a. The 'current-parent' mode

VI.b. Isolation via 'current' mode

VI.c. Deep nesting (Level 3)

VI.c.1. Deep Component A

VI.c.2. Deep navigation test

### VII. Style Customization

VII.a. The 3-state system

VII.b. Advanced Opacity & Inheritance

VII.b.1. The Float Shortcut (Clone & Fade)

VII.b.2. Partial Inheritance (Mix & Match)

### VIII. Customizable Markers

VIII.a. Static Marker

VIII.b. State-based Markers (Dictionary)

VIII.c. Dynamic Markers (Function)

VIII.d. Marker Alignment

### IX. The anti-jitter mechanism

IX.a. Colors and decorations

### X. Fine-grained spacing management

X.a. Inter-level spacing

X.b. Horizontal indentation

### XI. Numbering system

XI.a. Complex hierarchical formats

XI.b. Advanced textual prefixes

### XII. Filtering Content

XII.a. Label-based filtering

XII.b. Logic-based filtering

XII.c. Recursive filtering

### XIII. Advanced Behavior

XIII.a. Page-based matching

### XIV. Title Truncation

XIV.a. Global Truncation

XIV.b. Per-level Truncation

XIV.c. Horizontal Truncation

### XV. Short Titles

XV.a. Basic Usage

XV.b. Disabling Short Titles

### XVI. Additional information

## 11.2 Advanced textual prefixes

To use long words like "Chapter" without errors, pass a function. This prevents Typst from interpreting letters like 'a' or 'i' as numbering models.



```
show-numbering: true,  
numbering-format: (..n) => 'Chapter ' +  
numbering('1', ..n) + ' : '
```

### Secure 'Chapter' Prefix

Chapter 1 : Introduction

Chapter 2 : Global Configuration

Chapter 3 : Function documentation

Chapter 4 : Layout Modes

Chapter 5 : Navigation & Interactivity

Chapter 6 : Visibility

Chapter 7 : Style Customization

Chapter 8 : Customizable Markers

Chapter 9 : The anti-jitter mechanism

Chapter 10 : Fine-grained spacing management

**Chapter 11 : Numbering system**

Chapter 12 : Filtering Content

Chapter 13 : Advanced Behavior

Chapter 14 : Title Truncation

Chapter 15 : Short Titles

Chapter 16 : Additional information

## 12 Filtering Content

The `filter` parameter allows you to programmatically include or exclude headings from the outline. It expects a callback function (`heading`) => `boolean`.

The heading object passed to the filter contains standard properties (`level`, `body`, `label`, `counter`) as well as context properties: `parent-h1` and `parent-h2`.

### 12.1 Label-based filtering

In this document, the current section “Filtering Content” has been tagged with the label `<hidden>`.

```
progressive-outline(level-2-mode: 'none')
```

#### Standard Outline (No Filter)

- 1 Introduction
- 2 Global Configuration
- 3 Function documentation
- 4 Layout Modes
- 5 Navigation & Interactivity
- 6 Visibility
- 7 Style Customization
- 8 Customizable Markers
- 9 The anti-jitter mechanism
- 10 Fine-grained spacing management
- 11 Numbering system
- 12 Filtering Content**
- 13 Advanced Behavior
- 14 Title Truncation
- 15 Short Titles
- 16 Additional information

```
progressive-outline(  
  level-2-mode: 'none',  
  filter: h => h.label !== <hidden>  
)
```

#### Filtered Outline (Label)

- 1 Introduction
- 2 Global Configuration
- 3 Function documentation
- 4 Layout Modes
- 5 Navigation & Interactivity
- 6 Visibility
- 7 Style Customization
- 8 Customizable Markers
- 9 The anti-jitter mechanism
- 10 Fine-grained spacing management
- 11 Numbering system
- 13 Advanced Behavior
- 14 Title Truncation
- 15 Short Titles
- 16 Additional information

## 12.2 Logic-based filtering

You can also filter based on any heading property. Here, we filter the list to **keep only** the section named “Visibility”.

```
progressive-outline(
  level-2-mode: 'none',
  // Keep only the heading named
  'Visibility'
  filter: h => h.body == [Visibility]
)
```

## Filtered Outline (Content)

### 6 Visibility

Here, we create a custom rule: show all Level 1 headings, but show Level 2 headings **only** if they belong to the “Visibility” section.

```
progressive-outline(
  level-2-mode: 'all',
  filter: h => h.level == 1 or
    (h.level == 2 and h.parent-h1.body ==
    [Visibility])
)
```

## Conditional Depth

### 1 Introduction

### 2 Global Configuration

### 3 Function documentation

### 4 Layout Modes

### 5 Navigation & Interactivity

### 6 Visibility

#### 6.1 The ‘current-parent’ mode

#### 6.2 Isolation via ‘current’ mode

#### 6.3 Deep nesting (Level 3)

### 7 Style Customization

### 8 Customizable Markers

### 9 The anti-jitter mechanism

### 10 Fine-grained spacing management

### 11 Numbering system

### 12 Filtering Content

### 13 Advanced Behavior

### 14 Title Truncation

### 15 Short Titles

### 16 Additional information

## 12.3 Recursive filtering

The filtering logic is recursive: if a parent heading (e.g., a Section) is excluded by the filter, all its children (Subsections and Sub-subsections) are automatically hidden as well, even if they would have passed the filter individually.

```
// Hiding a parent automatically hides its
children
progressive-outline(
  level-2-mode: 'all',
  filter: h => h.label != <hidden>
)
```

## Recursive Hiding

### 1 Introduction

### 2 Global Configuration

### 3 Function documentation

### 4 Layout Modes

- 4.1 Vertical Layout
- 4.2 Horizontal Layout

### 5 Navigation & Interactivity

### 6 Visibility

- 6.1 The 'current-parent' mode
- 6.2 Isolation via 'current' mode
- 6.3 Deep nesting (Level 3)

### 7 Style Customization

- 7.1 The 3-state system
- 7.2 Advanced Opacity & Inheritance

### 8 Customizable Markers

- 8.1 Static Marker
- 8.2 State-based Markers (Dictionary)
- 8.3 Dynamic Markers (Function)
- 8.4 Marker Alignment

### 9 The anti-jitter mechanism

- 9.1 Colors and decorations

### 10 Fine-grained spacing management

- 10.1 Inter-level spacing
- 10.2 Horizontal indentation

### 11 Numbering system

- 11.1 Complex hierarchical formats
- 11.2 Advanced textual prefixes

### 13 Advanced Behavior

- 13.1 Page-based matching

### 14 Title Truncation

- 14.1 Global Truncation
- 14.2 Per-level Truncation
- 14.3 Horizontal Truncation

### 15 Short Titles

- 15.1 Basic Usage
- 15.2 Disabling Short Titles

### 16 Additional information

## 13 Advanced Behavior

### 13.1 Page-based matching

In contexts like sidebars, the outline is rendered in the page margin or background before the slide content. This can cause the active heading detection to fail because the content is technically “after” the sidebar in the document flow.

Setting `match-page-only: true` solves this by considering any heading on the current page as “active”, ignoring precise vertical positioning.

match-page-only: true

## Sidebar Logic

- 1 Introduction
- 2 Global Configuration
- 3 Function documentation
- 4 Layout Modes
- 5 Navigation & Interactivity
- 6 Visibility
- 7 Style Customization
- 8 Customizable Markers
- 9 The anti-jitter mechanism
- 10 Fine-grained spacing management
- 11 Numbering system
- 12 Filtering Content
- 13 Advanced Behavior
- 14 Title Truncation**
- 15 Short Titles
- 16 Additional information

## 14 Title Truncation

For very long titles that might break the layout (especially in sidebars or breadcrumbs), you can use the `max-length` parameter to automatically truncate the text.

### 14.1 Global Truncation

Setting `max-length` to an integer applies the limit to all heading levels.

```
progressive-outline(  
  max-length: 20  
)
```

## Global Truncation (Max 20 chars)

- 1 Introduction
- 2 Global Configuration
- 3 Function documentati...
- 4 Layout Modes
- 5 Navigation & Interac...
- 6 Visibility
- 7 Style Customization
- 8 Customizable Markers
- 9 The anti-jitter mech...
- 10 Fine-grained spacing...
- 11 Numbering system
- 12 Filtering Content
- 13 Advanced Behavior
- 14 Title Truncation**
  - 14.1 Global Truncation
  - 14.2 Per-level Truncation
  - 14.3 Horizontal Truncatio...
- 15 Short Titles
- 16 Additional informati...

## 14.2 Per-level Truncation

You can also pass a dictionary to define different limits for each level.

```
progressive-outline(  
  max-length: (level-1: 10, level-2: 30)  
)
```

## Granular Truncation

- 1 Introducti...
- 2 Global Con...
- 3 Function d...
- 4 Layout Mod...
- 5 Navigation...
- 6 Visibility
- 7 Style Cust...
- 8 Customizab...
- 9 The anti-j...
- 10 Fine-grain...
- 11 Numbering ...
- 12 Filtering ...
- 13 Advanced B...
- 14 Title Trun...
  - 14.1 Global Truncation
  - 14.2 Per-level Truncation
  - 14.3 Horizontal Truncation
- 15 Short Titl...
- 16 Additional...

## 14.3 Horizontal Truncation

This is particularly useful for horizontal breadcrumbs to ensure they don't overflow the page width.

```
progressive-outline(  
  layout: 'horizontal',  
  max-length: 15,  
  separator: ' > '  
)
```

## Truncated Breadcrumb

1 Introduction > 2 Global Configur... > 3 Function docume... > 4 Lay

## 15 Short Titles

Sometimes automatic truncation is not enough, and you want to display a completely different (shorter) text in the outline while keeping the full descriptive title in the content.

You can define a short title by placing a `#metadata("...") <short>` element immediately after the heading.

### 15.1 Basic Usage

By default, `use-short-title` is `false`. To use your manual short titles, you must set it to `true`.

```
progressive-outline(  
  use-short-title: true,  
  level-1-mode: 'current',  
  level-2-mode: 'current-parent',  
  level-3-mode: 'current-parent'  
)
```

## Using Manual Short Titles

- 15 Short
  - 15.1 Example
  - 15.2 Configuration

### 15.2 Disabling Short Titles

If you want to ignore the metadata and force the original titles, keep `use-short-title: false` (the default).

```
progressive-outline(  
  level-1-mode: 'current',  
  level-2-mode: 'current-parent',  
  level-3-mode: 'current-parent'  
)
```

## Original Titles (Default)

### 15 Short Titles

15.1 Basic Usage

15.2 Disabling Short Titles

You can also control this per-level: `use-short-title: (level-1: true, level-2: false)`

## 16 Additional information

It is optimized to work within presentation themes (like `progressive-outline`), but can be used in any standard Typst document.