

# Guide: Progressive Outline

## 1 Function documentation

This section details all the parameters available for the progressive-outline function.

Option	Type	Effect & Expected Values
level-X-mode	string	Defines the visibility of level X (1, 2, or 3). Values: "all", "current", "current-parent", "none".
text-styles	dict	Dictionary of Typst text(...) styles for each level. Structure: (level-X: (active: (...), completed: (...), inactive: (...))).
spacing	dict	Controls vertical space (v-between-X-Y) and horizontal indentation (indent-X) between elements.
show-numbering	bool	Enables or disables the display of heading numbering.
numbering-format	str   func	Typst numbering format (e.g., "1.1") or custom function (.n) => ...
match-page-only	bool	If true, considers a heading active if it is on the same page, regardless of its Y position. Useful for sidebars.
filter	func	A callback function (heading) => bool to programmatically include or exclude headings.

## 2 Visibility

This section covers the level-X-mode parameters.

### 2.1 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

##### Function documentation

##### Visibility

- The ‘current-parent’ mode
- Isolation via ‘current’ mode
- Deep nesting (Level 3)

##### Style Customization

##### Fine-grained spacing management

##### Numbering system

##### Filtering Content

##### Advanced Behavior

##### Additional information

## 2.2 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

#### Visibility

## 2.3 Deep nesting (Level 3)

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

### 2.3.1 Deep Component A

#### 2.3.2 Deep navigation test

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

### Level 3 Siblings

#### Function documentation

##### Visibility

The 'current-parent' mode  
Isolation via 'current' mode  
**Deep nesting (Level 3)**  
Deep Component A  
**Deep navigation test**

##### Style Customization

The 3-state system  
The anti-jitter mechanism  
Colors and decorations

##### Fine-grained spacing management

Inter-level spacing  
Horizontal indentation

##### Numbering system

Complex hierarchical formats  
Advanced textual prefixes

##### Filtering Content

Label-based filtering  
Logic-based filtering  
Recursive filtering

##### Advanced Behavior

Page-based matching

##### Additional information

## 3 Style Customization

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

### 3.1 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

Function documentation

Visibility

### Style Customization

Fine-grained spacing management

Numbering system

Filtering Content

Advanced Behavior

Additional information

## 3.2 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

Function documentation

Visibility

### Style Customization

Fine-grained spacing management

Numbering system

Filtering Content

Advanced Behavior

Additional information

## 3.3 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

### Function documentation

#### Visibility

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

#### Style Customization

The 3-state system  
The anti-jitter mechanism  
*Colors and decorations*

### Fine-grained spacing management

Inter-level spacing  
Horizontal indentation

#### Numbering system

Complex hierarchical formats  
Advanced textual prefixes

#### Filtering Content

Label-based filtering  
Logic-based filtering  
Recursive filtering

#### Advanced Behavior

Page-based matching

#### Additional information

## 4 Fine-grained spacing management

The spacing dictionary sculpts the rhythm.

### 4.1 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

### Function documentation

#### Visibility

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

#### Style Customization

The 3-state system  
The anti-jitter mechanism  
Colors and decorations

### Fine-grained spacing management

#### Inter-level spacing

Horizontal indentation

#### Numbering system

Complex hierarchical formats  
Advanced textual prefixes

#### Filtering Content

Label-based filtering  
Logic-based filtering  
Recursive filtering

#### Advanced Behavior

Page-based matching

#### Additional information

## 4.2 Horizontal indentation

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

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

## Marked Indentation

### Function documentation

#### Visibility

- The 'current-parent' mode
- Isolation via 'current' mode
- Deep nesting (Level 3)
  - Deep Component A
  - Deep navigation test

#### Style Customization

- The 3-state system
- The anti-jitter mechanism
- Colors and decorations

#### Fine-grained spacing management

- Inter-level spacing

#### Horizontal indentation

#### Numbering system

- Complex hierarchical formats
- Advanced textual prefixes

#### Filtering Content

- Label-based filtering
- Logic-based filtering
- Recursive filtering

#### Advanced Behavior

- Page-based matching

#### Additional information

## 5 Numbering system

The function relies on Typst's native engine.

### 5.1 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. Function documentation

### II. Visibility

- II.a. The 'current-parent' mode
- II.b. Isolation via 'current' mode
- II.c. Deep nesting (Level 3)
  - II.c.1. Deep Component A
  - II.c.2. Deep navigation test

### III. Style Customization

- III.a. The 3-state system
- III.b. The anti-jitter mechanism
- III.c. Colors and decorations

### IV. Fine-grained spacing management

- IV.a. Inter-level spacing
- IV.b. Horizontal indentation

### V. Numbering system

- V.a. Complex hierarchical formats
- V.b. Advanced textual prefixes

### VI. Filtering Content

- VI.a. Label-based filtering
- VI.b. Logic-based filtering
- VI.c. Recursive filtering

### VII. Advanced Behavior

- VII.a. Page-based matching

### VIII. Additional information

## 5.2 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 : Function documentation

Chapter 2 : Visibility

Chapter 3 : Style Customization

Chapter 4 : Fine-grained spacing management

Chapter 5 : Numbering system

Chapter 6 : Filtering Content

Chapter 7 : Advanced Behavior

Chapter 8 : Additional information

## 6 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`.

## 6.1 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)

Function documentation

Visibility

Style Customization

Fine-grained spacing management

Numbering system

**Filtering Content**

**Advanced Behavior**

**Additional information**

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

### Filtered Outline (Label)

Function documentation

Visibility

Style Customization

Fine-grained spacing management

Numbering system

**Advanced Behavior**

**Additional information**

## 6.2 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)

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

### Function documentation

#### Visibility

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

#### Style Customization

#### Fine-grained spacing management

#### Numbering system

#### Filtering Content

#### Advanced Behavior

#### Additional information

## 6.3 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

### Function documentation

#### Visibility

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

#### Style Customization

The 3-state system  
The anti-jitter mechanism  
Colors and decorations

#### Fine-grained spacing management

Inter-level spacing  
Horizontal indentation

#### Numbering system

Complex hierarchical formats  
Advanced textual prefixes

#### Advanced Behavior

Page-based matching

#### Additional information

## 7 Advanced Behavior

### 7.1 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

Function documentation

Visibility

Style Customization

Fine-grained spacing management

Numbering system

Filtering Content

Advanced Behavior

**Additional information**

## 8 Additional information

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