Component Documentation

Popup

The component provides a highly flexible floating container for displaying contextual content such as tooltips, menus, dropdowns, or custom overlays. It supports multiple trigger types, advanced positioning, animation, and full accessibility compliance. Use it to create interactive overlays that appear on click, hover, focus, or programmatically.

How to use

import { PopupComponent } from "@aava/play-core" ;

Basic Usage

A simple popup triggered by a button click, displaying custom content.

```
// app.component.html
<button aava-button (click)="isOpen = !isOpen">Toggle Popup</button>
<aava-popup [isOpen]="isOpen" (closed)="isOpen = false">
  <div style="padding: 1rem;">Hello from Popup!</div>
</aava-popup>
```

Positioning

Popup supports multiple placement options relative to the trigger element:

- Top
- Bottom
- Left
- Right
- Auto (smart positioning)

```
// app.component.html
<button aava-button>Top</button>
<aava-popup placement="top">
  <div style="padding: 1rem;">Top Placement</div>
</aava-popup>

<button aava-button>Bottom</button>
<aava-popup placement="bottom">
  <div style="padding: 1rem;">Bottom Placement</div>
</aava-popup>

<button aava-button>Left</button>
<aava-popup placement="left">
  <div style="padding: 1rem;">Left Placement</div>
</aava-popup>

<button aava-button>Right</button>
<aava-popup placement="right">
  <div style="padding: 1rem;">Right Placement</div>
</aava-popup>

<button aava-button>Auto</button>
<aava-popup placement="auto">
  <div style="padding: 1rem;">Auto Placement</div>
</aava-popup>
```

Custom Content

Render any custom content inside the popup, including forms, lists, or interactive elements.

```
// app.component.html
<button aava-button>Show Custom Content</button>
<aava-popup>
  <form style="padding: 1rem;">
    <label>Name: <input type="text" /></label><br />
    <button type="submit" aava-button>Submit</button>
  </form>
</aava-popup>
```

Accessibility

Popup is fully accessible:

• Keyboard navigation
• ARIA roles and attributes
• Focus management
• Dismiss on Escape

API Reference

Inputs

Property	Type	Default	Description
isOpen	boolean	false	Whether the popup is open
trigger	'click' | 'hover' | 'focus' | 'manual'	'click'	Trigger type for showing the popup
placement	'top' | 'bottom' | 'left' | 'right' | 'auto'	'bottom'	Popup placement relative to trigger
offset	number	8	Offset distance from trigger (px)
closeOnClickOutside	boolean	true	Close popup when clicking outside
closeOnEscape	boolean	true	Close popup on Escape key
disabled	boolean	false	Disable the popup
backdrop	boolean	false	Show a backdrop behind the popup
zIndex	number	1000	z-index for popup layering
animation	'fade' | 'scale' | 'none'	'fade'	Animation style for popup
containerClass	string	''	Custom CSS class for popup container
containerStyle	Record	{}	Custom styles for popup container

Outputs

Event	Type	Description
opened	EventEmitter	Emitted when popup is opened
closed	EventEmitter	Emitted when popup is closed
positioned	EventEmitter	Emitted when popup is positioned

Methods

Method	Parameters	Description
open()	void	Open the popup programmatically
close()	void	Close the popup programmatically
toggle()	void	Toggle popup open/close state
reposition()	void	Recalculate popup position

CSS Custom Properties

Property	Description
--popup-background	Background color of the popup
--popup-border-radius	Border radius of the popup
--popup-box-shadow	Box shadow for the popup
--popup-z-index	z-index for popup layering
--popup-animation-duration	Duration of the popup animation
--popup-arrow-size	Size of the popup arrow (if present)
--popup-arrow-color	Color of the popup arrow
--popup-backdrop-background	Background color for the backdrop

Accessibility Guidelines

- Keyboard Navigation : Tab, Shift+Tab, Escape to close
- ARIA Roles : role="dialog" or role="tooltip" as appropriate
- Focus Management : Focus is trapped within popup when open
- Dismissal : Popup closes on Escape or outside click
- Screen Reader Support : Popup content is announced

Best Practices

- Use the appropriate trigger for your use case (click for menus, hover for tooltips)
- Always provide accessible labels and ARIA attributes
- Avoid using popups for critical information that must not be missed
- Test keyboard and screen reader interactions
- Use smart positioning to avoid clipping or overflow
- Keep popup content concise and focused