Component Documentation

Checkbox

The component provides a highly interactive checkbox element with sophisticated animations, multiple visual variants, and comprehensive state management. It supports standard checked/unchecked states, indeterminate state for partial selections, and full accessibility compliance.

How to use

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

Basic Usage

Simple checkbox implementation with default settings and label.

```
<aava-checkbox
  label="Accept terms and conditions"
  [isChecked]="false"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>
<aava-checkbox
  label="Subscribe to newsletter"
  [isChecked]="false"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>
<aava-checkbox
  label="Enable notifications"
  [isChecked]="false"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>

---

onCheckboxChange(checked: boolean) {
    console.log('Checkbox changed:', checked);
  }
```

Variants

The checkbox component supports 3 distinct animation variants, each with unique visual feedback and timing.

```
<aava-checkbox
  label="Default Checked"
  variant="default"
  [isChecked]="true"
></aava-checkbox>
<aava-checkbox
  label="With-bg Checked"
  variant="with-bg"
  [isChecked]="true"
></aava-checkbox>
<aava-checkbox
  label="Animated Checked"
  variant="animated"
  [isChecked]="true"
></aava-checkbox>
```

Available Variants

• default - Clean checkbox with smooth draw/erase animations (250ms draw, 300ms erase)
• with-bg - Background fill effect with fast transitions (150ms animations)
• animated - Complex scaling background animation with staggered timing (300ms background + 300ms delay + checkmark)

Orientations

Flexible layout options for different design requirements and form arrangements.

```
<div>
  <h1>Vertical Layout (Default)</h1>
  <div>
    <aava-checkbox
      alignment="vertical"
      label="Development Tools"
      [isChecked]="false"
    ></aava-checkbox>
    <aava-checkbox
      alignment="vertical"
      label="Design Software"
      [isChecked]="true"
    ></aava-checkbox>
    <aava-checkbox
      alignment="vertical"
      label="Project Management"
      [isChecked]="false"
    ></aava-checkbox>
    <aava-checkbox
      alignment="vertical"
      label="Communication Apps"
      [isChecked]="true"
    ></aava-checkbox>
  </div>

  <h1>Horizontal Layout</h1>
  <div>
    <aava-checkbox
      alignment="horizontal"
      label="React"
      [isChecked]="true"
    ></aava-checkbox>
    <aava-checkbox
      alignment="horizontal"
      label="Angular"
      [isChecked]="false"
    ></aava-checkbox>
    <aava-checkbox
      alignment="horizontal"
      label="Vue.js"
      [isChecked]="true"
    ></aava-checkbox>
    <aava-checkbox
      alignment="horizontal"
      label="Svelte"
      [isChecked]="false"
    ></aava-checkbox>
    <aava-checkbox
      alignment="horizontal"
      label="Next.js"
      [isChecked]="false"
    ></aava-checkbox>
    <aava-checkbox
      alignment="horizontal"
      label="Nuxt.js"
      [isChecked]="false"
    ></aava-checkbox>
  </div>
</div>
```

Available Orientations

• vertical - Default stacked layout with checkboxes arranged vertically
• horizontal - Inline layout with checkboxes arranged horizontally side by side

Sizes

Three size options to accommodate different interface densities and visual hierarchies.

```
<aava-checkbox label="Small" size="sm" variant="default" [isChecked]="false">
</aava-checkbox>
<aava-checkbox label="Medium" size="md" variant="default" [isChecked]="true">
</aava-checkbox>
<aava-checkbox label="Large" size="lg" variant="default" [isChecked]="false">
</aava-checkbox>
```

Available Sizes

• sm (Small) - 16x16px checkbox for compact interfaces
• md (Medium) - 20x20px checkbox for standard layouts (default)
• lg (Large) - 24x24px checkbox for prominent placements

States

Comprehensive state management including checked, unchecked, disabled, and indeterminate states.

```
<aava-checkbox label="Unchecked" [isChecked]="false"> </aava-checkbox>
<aava-checkbox label="Checked" [isChecked]="true"> </aava-checkbox>
<aava-checkbox label="Disabled Unchecked" [disable]="true" [isChecked]="false">
</aava-checkbox>
<aava-checkbox label="Disabled Checked" [disable]="true" [isChecked]="true">
</aava-checkbox>
<aava-checkbox
  label="Disabled Indeterminate"
  [disable]="true"
  [indeterminate]="true"
>
</aava-checkbox>
<aava-checkbox label="Indeterminate" [indeterminate]="true"> </aava-checkbox>
<aava-checkbox
  label="Indeterminate (With-bg)"
  variant="with-bg"
  [indeterminate]="true"
>
</aava-checkbox>
<aava-checkbox
  label="Indeterminate (Animated)"
  variant="animated"
  [indeterminate]="true"
>
</aava-checkbox>
```

Available States

• unchecked - Default empty state
• checked - Selected state with checkmark animation
• disabled - Non-interactive state (can be checked or unchecked)
• indeterminate - Partial selection state with horizontal line indicator

State Behavior

Checked State:

• Displays animated checkmark
• Emits true value on change
• Visual feedback varies by variant

Disabled State:

• Non-interactive (no click/keyboard events)
• Dimmed appearance with disabled styling
• Maintains current checked/unchecked state
• Custom disabled colors via CSS tokens

Indeterminate State:

• Shows horizontal line instead of checkmark
• Used for "select all" scenarios in groups
• Clicking transitions to fully checked state
• Emits true when transitioning from indeterminate

Indeterminate State

Special state for representing partial selections in checkbox groups or tree structures.

```
<div class="checkbox-hierarchy">
  <aava-checkbox
    label="Select All Tasks"
    [isChecked]="parentChecked"
    [indeterminate]="indeterminate"
    (isCheckedChange)="onParentChanged($event)"
  >
  </aava-checkbox>

  <div class="child-checkboxes">
    <aava-checkbox
      *ngFor="let child of children; let i = index"
      [label]="child.label"
      [isChecked]="child.checked"
      (isCheckedChange)="onChildChanged(i, $event)"
    >
    </aava-checkbox>
  </div>
</div>

---

parentChecked = false;
  indeterminate = false
  children = [
    { label: 'Child task 1', checked: false },
    { label: 'Child task 2', checked: false },
    { label: 'Child task 3', checked: false },
  ];

  onParentChanged(checked: boolean): void {
    this.parentChecked = checked;
    this.indeterminate = false;
    this.children = this.children.map((child) => ({ ...child, checked }));
  }

  onChildChanged(index: number, checked: boolean): void {
    this.children[index].checked = checked;
    this.updateParentState();
  }

  updateParentState(): void {
    const total = this.children.length;
    const checkedCount = this.children.filter((c) => c.checked).length;

    if (checkedCount === total) {
      this.parentChecked = true;
      this.indeterminate = false;
    } else if (checkedCount === 0) {
      this.parentChecked = false;
      this.indeterminate = false;
    } else {
      this.parentChecked = false;
      this.indeterminate = true;
    }
  }


  updateParentStates(items: any[]): void {
    items.forEach((item) => {
```

```
      if (item.children) {
        this.updateParentStates(item.children);
        const total = item.children.length;
        const checkedCount = item.children.filter(
          (c: any) => c.checked || c.indeterminate
        ).length;
        const fullyCheckedCount = item.children.filter(
          (c: any) => c.checked && !c.indeterminate
        ).length;

        if (fullyCheckedCount === total) {
          item.checked = true;
          item.indeterminate = false;
        } else if (checkedCount === 0) {
          item.checked = false;
          item.indeterminate = false;
        } else {
          item.checked = false;
          item.indeterminate = true;
        }
      }
    });
  }
```

Indeterminate Usage

The indeterminate state is particularly useful for:

- Select All checkboxes - When some but not all items are selected
- Tree structures - Parent nodes with partially selected children
- Group controls - Representing mixed states in checkbox groups
- Bulk actions - Indicating partial selection in data tables

Indeterminate Behavior

- Displays horizontal line indicator instead of checkmark
- Clicking indeterminate checkbox transitions to fully checked
- Cannot be set via user interaction (must be programmatically controlled)
- Supports all variants and sizes
- Maintains proper ARIA attributes (aria-checked="mixed")

Labels & Accessibility

Comprehensive accessibility features including keyboard navigation, ARIA attributes, and label support.

```
<aava-checkbox
  label="First checkbox (Tab to focus)"
  [(isChecked)]="keyboardStates.first"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>
<aava-checkbox
  label="Second checkbox (Space/Enter to toggle)"
  [(isChecked)]="keyboardStates.second"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>
<aava-checkbox
  label="Third checkbox (Try keyboard navigation)"
  [(isChecked)]="keyboardStates.third"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>
<aava-checkbox
  label="Standard checkbox with ARIA"
  [(isChecked)]="ariaStates.standard"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>
<aava-checkbox
  label="Indeterminate checkbox (aria-checked='mixed')"
  [indeterminate]="true"
>
</aava-checkbox>
<aava-checkbox
  label="Disabled checkbox (aria-disabled='true')"
  [disable]="true"
  [isChecked]="true"
>
</aava-checkbox>
<aava-checkbox
  label="Checkbox with visible label"
  [(isChecked)]="labelStates.visible"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>
<aava-checkbox
  label="Clickable label (clicking label toggles checkbox)"
  [(isChecked)]="labelStates.clickable"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>

---

keyboardStates = {
    first: false,
    second: true,
    third: false,
  };

  ariaStates = {
    standard: false,
  };
```

```
labelStates = {
  visible: true,
  noVisible: false,
  clickable: false,
};

onCheckboxChange(checked: boolean) {
  console.log('Checkbox changed:', checked);
}
```

Accessibility Features

Keyboard Navigation:

• Space - Toggle checkbox state
• Enter - Toggle checkbox state
• Tab/Shift+Tab - Navigate to/from checkbox

ARIA Compliance:

• role="checkbox" - Semantic role identification
• aria-checked="true|false|mixed" - Current state indication
• aria-disabled="true|false" - Disabled state indication
• aria-label - Accessibility label (falls back to visible label)

Visual Accessibility:

• High contrast focus indicators
• Proper color contrast ratios via CSS tokens
• Scalable vector icons (no pixelation)
• Motion respects user preferences

Label Behavior

• Optional labels - Checkbox works with or without visible labels
• Clickable labels - Clicking label text toggles checkbox
• Accessible labels - Always provide meaningful labels for screen readers
• Flexible positioning - Label appears to the right of checkbox

Events

Event handling for checkbox state changes and user interactions.

```
<aava-checkbox
  label="Event Demo Checkbox"
  [isChecked]="false"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>
<aava-checkbox
  label="Another Event Checkbox"
  [isChecked]="false"
  (isCheckedChange)="onCheckboxChange($event)"
>
</aava-checkbox>

---

onCheckboxChange(checked: boolean) {
    console.log('Checkbox changed:', checked);
  }
```

Available Events

• isCheckedChange - Emitted when checkbox state changes (boolean value)

Event Details

isCheckedChange Event:

• Type: EventEmitter
• Emitted when: User clicks checkbox or uses keyboard to toggle
• Value: true when checked, false when unchecked
• Timing: Emitted after animation completes (timing varies by variant)
• Indeterminate: Emits true when transitioning from indeterminate to checked

Event Timing by Variant

• Default: Immediate for checking, 300ms delay for unchecking
• With-bg: 150ms delay for both checking and unchecking
• Animated: 600ms delay for checking, 300ms delay for unchecking

API Reference

Inputs

Property	Type	Default	Description
variant	'default' | 'with-bg' | 'animated'	'default'	Animation style variant
size	'sm' | 'md' | 'lg'	'md'	Checkbox size
alignment	'vertical' | 'horizontal'	'vertical'	Layout orientation for checkbox groups

Property	Type	Default	Description
label	string	''	Visible label text
isChecked	boolean	false	Whether checkbox is checked
indeterminate	boolean	false	Whether checkbox is in indeterminate state
disable	boolean	false	Whether checkbox is disabled
customStyles	Record	{}	CSS custom properties override
error	string	''	Error message text
id	string	''	Unique checkbox identifier

Outputs

Event	Type	Description
isCheckedChange	EventEmitter	Emitted when checkbox state changes

Properties

Property	Type	Description
showIcon	boolean	Whether to show the icon (checkmark or indeterminate)
showCheckmark	boolean	Whether to show checkmark specifically

CSS Custom Properties

Property	Description
--checkbox-box-background	Background color of the checkbox box
--checkbox-box-checked-border	Border color when checked
--checkbox-box-checked-background	Background color when checked
--checkbox-box-border-disabled	Border color when disabled

Property	Description
--checkbox-box-background-disabled	Background color when disabled
--checkbox-icon-color-disabled	Icon color when disabled
--checkbox-box-checked-color	Checkmark color when checked
--checkbox-label-color	Text color for the label
--checkbox-label-color-disabled	Label text color when disabled
--checkbox-label-cursor	Cursor style for clickable label
--checkbox-label-cursor-disabled	Cursor style when disabled
--checkbox-cursor-disabled	Cursor style for disabled checkbox
--checkbox-size-small	Size dimensions for small variant
--checkbox-size-medium	Size dimensions for medium variant
--checkbox-size-large	Size dimensions for large variant

Animation Timing

Each variant has carefully tuned animation timing for optimal user experience:

Default Variant

• Check: 250ms cubic-bezier(0.25, 0.46, 0.45, 0.94)
• Uncheck: 300ms cubic-bezier(0.55, 0.06, 0.68, 0.19)

With-bg Variant

• Check: 150ms cubic-bezier(0.25, 0.46, 0.45, 0.94)
• Uncheck: 150ms cubic-bezier(0.55, 0.06, 0.68, 0.19)

Animated Variant

• Background: 300ms ease-out
• Checkmark: 300ms cubic-bezier(0.25, 0.46, 0.45, 0.94) with 300ms delay
• Uncheck: 300ms background + 150ms checkmark (simultaneous)

Best Practices

Design Guidelines

• Choose appropriate variants - Use default for most cases, with-bg for high contrast needs, animated for engaging interactions
• Size consistently - Match checkbox size to surrounding form elements and text size
• Group related checkboxes - Use proper form structure and fieldsets for related options
• Handle indeterminate properly - Use for partial selections in groups, not individual checkboxes

Accessibility

• Label everything - Always provide meaningful labels for accessibility
• Keyboard navigation - Ensure full keyboard support with proper focus management
• ARIA compliance - Use proper ARIA attributes for screen readers
• Visual accessibility - Maintain high contrast and clear focus indicators
• Motion preferences - Consider users who prefer reduced motion

Performance

• Respect motion preferences - Consider users who prefer reduced motion
• Optimize animations - Use CSS transforms and opacity for smooth performance
• Batch state changes - Avoid rapid successive state changes
• Use appropriate variants - Choose variants based on performance requirements