Menu

A versatile menu for navigation.

When To Use

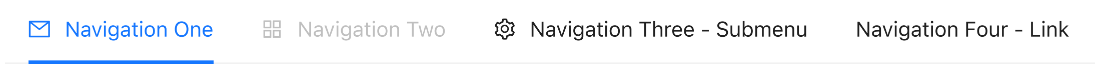
Navigation is an important part of any website, as a good navigation setup allows users to move around the site quickly and efficiently. Ant Design offers two navigation options: top and side. Top navigation provides all the categories and functions of the website. Side navigation provides the multi-level structure of the website.

More layouts with navigation: Layout.

Notes for developers

- Menu is rendered as a `ul` element, so it only supports `li` and `script-supporting` elements as children nodes。Your customized node should be wrapped by `Menu.Item`.
- Menu needs to collect its node structure, so its children should be `Menu.*` or encapsulated HOCs.

Examples



Top Navigation

Horizontal top navigation menu.

```
import React, { useState } from 'react';
import { AppstoreOutlined, MailOutlined, SettingOutlined } from '@ant-design/icons';
import type { MenuProps } from 'antd';
import { Menu } from 'antd';

const items: MenuProps['items'] = [
  {
    label: 'Navigation One',
    key: 'mail',
    icon: <MailOutlined />,
  },
  {
    label: 'Navigation Two',
    key: 'app',
    icon: <AppstoreOutlined />,
    disabled: true,
  },
  {
    label: 'Navigation Three - Submenu',
    key: 'SubMenu',
    icon: <SettingOutlined />,
    children: [
      {
        type: 'group',
        label: 'Item 1',
        children: [
          {
            label: 'Option 1',
            key: 'setting:1',
          },
          {
            label: 'Option 2',
            key: 'setting:2',
          },
        ],
      },
      {
        type: 'group',
        label: 'Item 2',
        children: [
          {
            label: 'Option 3',
            key: 'setting:3',
          },
          {
            label: 'Option 4',
            key: 'setting:4',
          },
        ],
      },
    ],
  },
  {
    label: (
      <a href="https://ant.design" target="_blank" rel="noopener noreferrer">
        Navigation Four - Link
      </a>
    ),
    key: 'alipay',
  },
];

const App: React.FC = () => {
```

Navigation One

Item 1

Option 1

Option 2

Item 2

Option 3

Option 4

Navigation Two

Navigation Three

Group

Option 13

Option 14

Inline menu

Vertical menu with inline submenus.

```
import React from 'react';
import { AppstoreOutlined, MailOutlined, SettingOutlined } from '@ant-design/icons';
import type { MenuProps } from 'antd';
import { Menu } from 'antd';

type MenuItem = Required<MenuProps>['items'][number];

function getItem(
  label: React.ReactNode,
  key: React.Key,
  icon?: React.ReactNode,
  children?: MenuItem[],
  type?: 'group',
): MenuItem {
  return {
    key,
    icon,
    children,
    label,
    type,
  } as MenuItem;
}

const items: MenuProps['items'] = [
  getItem('Navigation One', 'sub1', <MailOutlined />, [
    getItem('Item 1', 'g1', null, [getItem('Option 1', '1'), getItem('Option 2', '2')], 'group')
    getItem('Item 2', 'g2', null, [getItem('Option 3', '3'), getItem('Option 4', '4')], 'group')
  ]),

  getItem('Navigation Two', 'sub2', <AppstoreOutlined />, [
    getItem('Option 5', '5'),
    getItem('Option 6', '6'),
    getItem('Submenu', 'sub3', null, [getItem('Option 7', '7'), getItem('Option 8', '8')]),
  ]),

  { type: 'divider' },

  getItem('Navigation Three', 'sub4', <SettingOutlined />, [
```

Option 1

Option 2

Option 3

Navigation One

Option 5

Option 6

Option 7

Option 8

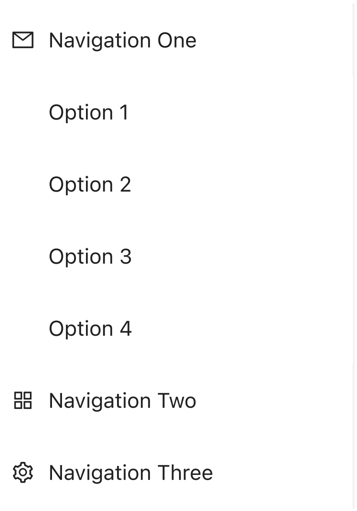
Navigation Two

Collapsed inline menu

Inline menu could be collapsed.

Here is a complete demo with sider layout.

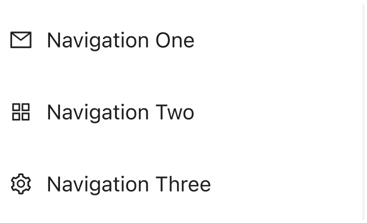
```
import React, { useState } from 'react';
import {
  AppstoreOutlined,
  ContainerOutlined,
  DesktopOutlined,
  MailOutlined,
  MenuFoldOutlined,
  MenuUnfoldOutlined,
  PieChartOutlined,
} from '@ant-design/icons';
import type { MenuProps } from 'antd';
import { Button, Menu } from 'antd';

type MenuItem = Required<MenuProps>['items'][number];

function getItem(
  label: React.ReactNode,
  key: React.Key,
  icon?: React.ReactNode,
  children?: MenuItem[],
  type?: 'group',
): MenuItem {
  return {
    key,
    icon,
    children,
    label,
    type,
  } as MenuItem;
}

const items: MenuItem[] = [
  getItem('Option 1', '1', <PieChartOutlined />),
  getItem('Option 2', '2', <DesktopOutlined />),
  getItem('Option 3', '3', <ContainerOutlined />),

  getItem('Navigation One', 'sub1', <MailOutlined />, [
    getItem('Option 5', '5'),
    getItem('Option 6', '6'),
    getItem('Option 7', '7')
```



Open current submenu only

Click the menu and you will see that all the other menus gets collapsed to keep the entire menu compact.

```
import React, { useState } from 'react';
import { AppstoreOutlined, MailOutlined, SettingOutlined } from '@ant-design/icons';
import type { MenuProps } from 'antd';
import { Menu } from 'antd';

type MenuItem = Required<MenuProps>['items'][number];

function getItem(
  label: React.ReactNode,
  key: React.Key,
  icon?: React.ReactNode,
  children?: MenuItem[],
  type?: 'group',
): MenuItem {
  return {
    key,
    icon,
    children,
    label,
    type,
  } as MenuItem;
}

const items: MenuItem[] = [
  getItem('Navigation One', 'sub1', <MailOutlined />, [
    getItem('Option 1', '1'),
    getItem('Option 2', '2'),
    getItem('Option 3', '3'),
    getItem('Option 4', '4'),
  ]),
  getItem('Navigation Two', 'sub2', <AppstoreOutlined />, [
    getItem('Option 5', '5'),
    getItem('Option 6', '6'),
    getItem('Submenu', 'sub3', null, [getItem('Option 7', '7'), getItem('Option 8', '8')]),
  ]),
  getItem('Navigation Three', 'sub4', <SettingOutlined />, [
    getItem('Option 9', '9'),
    getItem('Option 10', '10'),
    getItem('Option 11', '11'),
    getItem('Option 12', '12'),
  ]),
];

// submenu keys of first level
const rootSubmenuKeys = ['sub1', 'sub2', 'sub4'];

const App: React.FC = () => {
  const [openKeys, setOpenKeys] = useState(['sub1']);
```



Vertical menu

Submenus open as pop-ups.

```
import React from 'react';
import { AppstoreOutlined, MailOutlined, SettingOutlined } from '@ant-design/icons';
import type { MenuProps } from 'antd';
import { Menu } from 'antd';

type MenuItem = Required<MenuProps>['items'][number];

function getItem(
  label: React.ReactNode,
  key?: React.Key | null,
  icon?: React.ReactNode,
  children?: MenuItem[],
  type?: 'group',
): MenuItem {
  return {
    key,
    icon,
    children,
    label,
    type,
  } as MenuItem;
}

const items: MenuItem[] = [
  getItem('Navigation One', 'sub1', <MailOutlined />, [
    getItem('Item 1', null, null, [getItem('Option 1', '1'), getItem('Option 2', '2')], 'group')
    getItem('Item 2', null, null, [getItem('Option 3', '3'), getItem('Option 4', '4')], 'group')
  ]),

  getItem('Navigation Two', 'sub2', <AppstoreOutlined />, [
    getItem('Option 5', '5'),
    getItem('Option 6', '6'),
    getItem('Submenu', 'sub3', null, [getItem('Option 7', '7'), getItem('Option 8', '8')]),
  ]),

  getItem('Navigation Three', 'sub4', <SettingOutlined />, [
    getItem('Option 9', '9'),
    getItem('Option 10', '10'),
    getItem('Option 11', '11'),
    getItem('Option 12', '12'),
  ]),
];

const onClick: MenuProps['onClick'] = (e) => {
  console.log('click', e);
};

const App: React.FC = () => (
  <Menu onClick={onClick} style={{ width: 256 }} mode="vertical" items={items} />
);

export default App;
```

Dark

Navigation One

Option 1

Option 2

Option 3

Option 4

Navigation Two

Navigation Three

Menu Themes

There are two built-in themes: `light` and `dark`. The default value is `light`.

```
import React, { useState } from 'react';
import { AppstoreOutlined, MailOutlined, SettingOutlined } from '@ant-design/icons';
import type { MenuProps, MenuTheme } from 'antd';
import { Menu, Switch } from 'antd';

type MenuItem = Required<MenuProps>['items'][number];

function getItem(
  label: React.ReactNode,
  key?: React.Key | null,
  icon?: React.ReactNode,
  children?: MenuItem[],
  type?: 'group',
): MenuItem {
  return {
    key,
    icon,
    children,
    label,
    type,
  } as MenuItem;
}

const items: MenuItem[] = [
  getItem('Navigation One', 'sub1', <MailOutlined />, [
    getItem('Option 1', '1'),
    getItem('Option 2', '2'),
    getItem('Option 3', '3'),
    getItem('Option 4', '4'),
  ]),

  getItem('Navigation Two', 'sub2', <AppstoreOutlined />, [
    getItem('Option 5', '5'),
    getItem('Option 6', '6'),
    getItem('Submenu', 'sub3', null, [getItem('Option 7', '7'), getItem('Option 8', '8')]),
  ]),

  getItem('Navigation Three', 'sub4', <SettingOutlined />, [
    getItem('Option 9', '9'),
    getItem('Option 10', '10'),
    getItem('Option 11', '11'),
    getItem('Option 12', '12'),
  ]),
];

const App: React.FC = () => {
```

Light

Navigation One

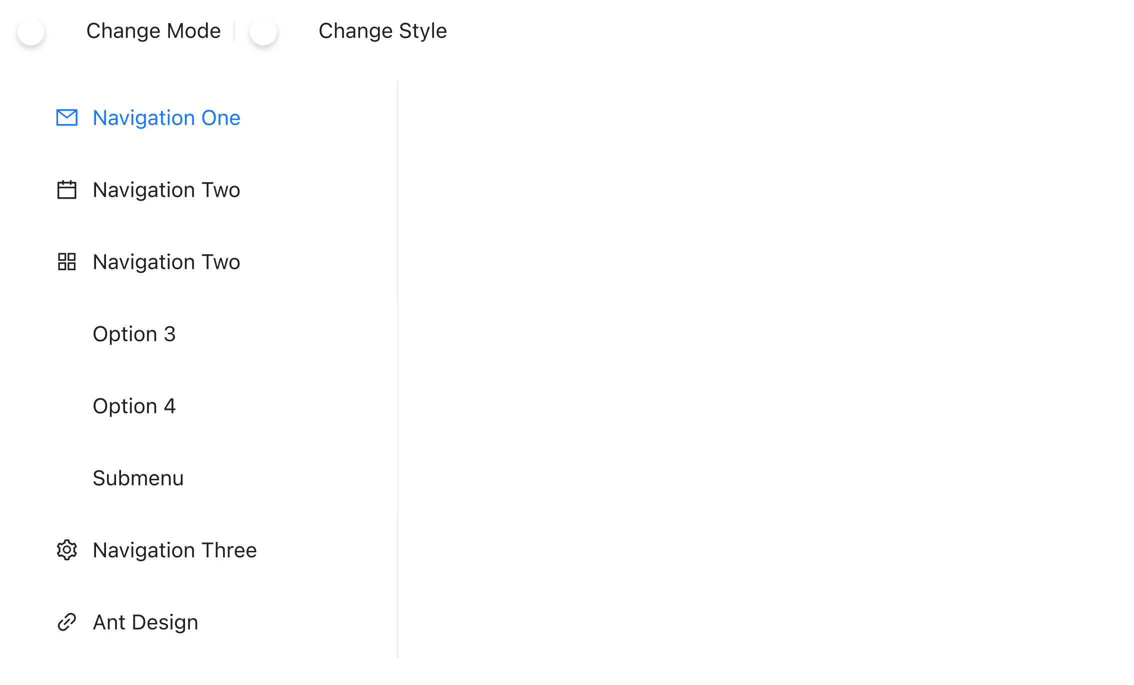
Option 5

Option 6

Sub-menu theme

You can config SubMenu theme with `theme` prop to enable different theme color effect. This sample is dark for root and light for SubMenu.

```
import React, { useState } from 'react';
import { MailOutlined } from '@ant-design/icons';
import type { MenuProps, MenuTheme } from 'antd';
import { Menu, Switch } from 'antd';

type MenuItem = Required<MenuProps>['items'][number];

function getItem(
  label: React.ReactNode,
  key?: React.Key | null,
  icon?: React.ReactNode,
  children?: MenuItem[],
  theme?: 'light' | 'dark',
): MenuItem {
  return {
    key,
    icon,
    children,
    label,
    theme,
  } as MenuItem;
}

const App: React.FC = () => {
  const [theme, setTheme] = useState<MenuTheme>('light');
  const [current, setCurrent] = useState('1');

  const changeTheme = (value: boolean) => {
    setTheme(value ? 'dark' : 'light');
  };

  const onClick: MenuProps['onClick'] = (e) => {
    setCurrent(e.key);
  };

  const items: MenuItem[] = [
    getItem(
      'Navigation One',
      'sub1',
      <MailOutlined />,
      [getItem('Option 1', '1'), getItem('Option 2', '2'), getItem('Option 3', '3')],
      theme,
    ),
    getItem('Option 5', '5'),
    getItem('Option 6', '6'),
  ];

  return (
    <>
      <Switch
        checked={theme === 'dark'}
        onChange={changeTheme}
        checkedChildren="Dark"
        unCheckedChildren="Light"
      />
```



Switch the menu type

Show the dynamic switching mode (between `inline` and `vertical`).

```
import React, { useState } from 'react';
import {
  AppstoreOutlined,
  CalendarOutlined,
  LinkOutlined,
  MailOutlined,
  SettingOutlined,
} from '@ant-design/icons';
import { Divider, Menu, Switch } from 'antd';
import type { MenuProps, MenuTheme } from 'antd/es/menu';

type MenuItem = Required<MenuProps>['items'][number];

function getItem(
  label: React.ReactNode,
  key?: React.Key | null,
  icon?: React.ReactNode,
  children?: MenuItem[],
): MenuItem {
  return {
    key,
    icon,
    children,
    label,
  } as MenuItem;
}

const items: MenuItem[] = [
  getItem('Navigation One', '1', <MailOutlined />),
  getItem('Navigation Two', '2', <CalendarOutlined />),
  getItem('Navigation Two', 'sub1', <AppstoreOutlined />, [
    getItem('Option 3', '3'),
    getItem('Option 4', '4'),
    getItem('Submenu', 'sub1-2', null, [getItem('Option 5', '5'), getItem('Option 6', '6')]),
  ]),
  getItem('Navigation Three', 'sub2', <SettingOutlined />, [
    getItem('Option 7', '7'),
    getItem('Option 8', '8'),
    getItem('Option 9', '9'),
    getItem('Option 10', '10'),
  ]),
  getItem(
    <a href="https://ant.design" target="_blank" rel="noopener noreferrer">
      Ant Design
      Ant Design
    </a>,
    'link',
    <LinkOutlined />,
  ),
];

const App: React.FC = () => {
  const [mode, setMode] = useState<'vertical' | 'inline'>('inline');
  const [theme, setTheme] = useState<MenuTheme>('light');

  const changeMode = (value: boolean) => {
    setMode(value ? 'vertical' : 'inline');
  };

  const changeTheme = (value: boolean) => {
    setTheme(value ? 'dark' : 'light');
  };

  return (
    <>
      <Switch onChange={changeMode} /> Change Mode
      <Divider type="vertical" />
      <Switch onChange={changeTheme} /> Change Style
      <br />
      <br />
      <Menu
        style={{ width: 256 }}
        defaultSelectedKeys={['1']}
        defaultOpenKeys={['sub1']}
        mode={mode}
        theme={theme}
        items={items}
      />
    </>
  );
};

export default App;
```

API

Menu

Param	Description	Type	Default value
defaultOpenKeys	Array with the keys of default opened sub menus	string[]	–
defaultSelectedKeys	Array with the keys of default selected menu items	string[]	–
expandIcon	custom expand icon of submenu	ReactNode | `(props: SubMenuProps & { isSubMenu: boolean }) => ReactNode`	–
forceSubMenuRender	Render submenu into DOM before it becomes visible	boolean	false
inlineCollapsed	Specifies the collapsed status when menu is inline mode	boolean	–
inlineIndent	Indent (in pixels) of inline menu items on each level	number	24
items	Menu item content	[ItemType[]]	–
mode	Type of menu	`vertical` | `horizontal` | `inline`	`vertical`
multiple	Allows selection of multiple items	boolean	false
openKeys	Array with the keys of currently opened sub-menus	string[]	–
overflowedIndicator	Customized the ellipsis icon when menu is collapsed horizontally	ReactNode	`<EllipsisOutlined />`
selectable	Allows selecting menu items	boolean	true
selectedKeys	Array with the keys of currently	string[]	–

Param	Description	Type	Default value
	selected menu items		
style	Style of the root node	CSSProperties	-
subMenuCloseDelay	Delay time to hide submenu when mouse leaves (in seconds)	number	0.1
subMenuOpenDelay	Delay time to show submenu when mouse enters, (in seconds)	number	0
theme	Color theme of the menu	`light` | `dark`	`light`
triggerSubMenuAction	Which action can trigger submenu open/close	`hover` | `click`	`hover`
onClick	Called when a menu item is clicked	function({ item, key, keyPath, domEvent })	-
onDeselect	Called when a menu item is deselected (multiple mode only)	function({ item, key, keyPath, selectedKeys, domEvent })	-
onOpenChange	Called when sub-menus are opened or closed	function(openKeys: string[])	-
onSelect	Called when a menu item is selected	function({ item, key, keyPath, selectedKeys, domEvent })	-

> More options in rc-menu

ItemType

> type ItemType = MenuItemType | SubMenuType | MenuItemGroupType | MenuDividerType;

MenuItemType

Param	Description	Type	Default value	Version
danger	Display the danger style	boolean	false	
disabled	Whether menu item is disabled	boolean	false	
icon	The icon of the menu item	ReactNode	-	
key	Unique ID of the menu item	string	-	
label	Menu label	ReactNode	-	

Param	Description	Type	Default value	Version
title	Set display title for collapsed item	string	-	

SubMenuType

Property	Description	Type	Default value	Version
children	Sub-menus or sub-menu items	ItemType[]	-	
disabled	Whether sub-menu is disabled	boolean	false	
icon	Icon of sub menu	ReactNode	-	
key	Unique ID of the sub-menu	string	-	
label	Menu label	ReactNode	-	
popupClassName	Sub-menu class name, not working when `mode="inline"`	string	-	
popupOffset	Sub-menu offset, not working when `mode="inline"`	[number, number]	-	
theme	Color theme of the SubMenu (inherits from Menu by default)		`light` | `dark`	-
onTitleClick	Callback executed when the sub-menu title is clicked	function({ key, domEvent })	-	

MenuItemGroupType

Define `type` as `group` to make as group:

```
const groupItem = {
  type: 'group', // Must have
  label: 'My Group',
  children: [],
};
```

Param	Description	Type	Default value	Version
children	Sub-menu items	MenuItemType[]	-	
label	The title of the group	ReactNode	-	

MenuDividerType

Divider line in between menu items, only used in vertical popup Menu or Dropdown Menu. Need define the `type` as `divider`：

```
const dividerItem = {
  type: 'divider', // Must have
};
```

Param	Description	Type	Default value	Version
dashed	Whether line is dashed	boolean	false	

FAQ

Why will Menu's children be rendered twice?

Menu collects structure info with twice-render to support HOC usage. Merging into one render may cause the logic to become much more complex. Contributions to help improve the collection logic are welcomed.

Why Menu do not responsive collapse in Flex layout?

Menu will render fully item in flex layout and then collapse it. You need tell flex not consider Menu width to enable responsive (online demo):

```
<div style={{ flex }}>
  <div style={{ ... }}>Some Content</div>
  <Menu style={{ minWidth: 0, flex: "auto" }} />
</div>
```

Design Token

▼ Global Token

Token Name	Description	Type	Default Value
colorBgContainer	Container background color, e.g: default button, input box, etc. Be sure not to confuse this with `colorBgElevated`.	`string`	□ #ffffff
colorBgElevated	Container background color of the popup layer, in dark mode the color value of this token will be a little brighter than `colorBgContainer`. E.g: modal, pop-up, menu, etc.	`string`	□ #ffffff
colorBgTextHover	Control the background color of text in hover state.	`string`	□ rgba(0, 0, 0, 0.06)
colorError	Used to represent the visual elements of the operation failure, such as the error Button, error Result component, etc.	`string`	□ #ff4d4f
colorErrorBg	The background color of the error state.	`string`	□ #fff2f0
colorErrorHover	The hover state of the error color.	`string`	□ #ff7875

Token Name	Description	Type	Default Value
colorFillAlter	Control the alternative background color of element.	`string`	□ `rgba(0, 0, 0, 0.02)`
colorFillContent	Control the background color of content area.	`string`	□ `rgba(0, 0, 0, 0.06)`
colorPrimary	Brand color is one of the most direct visual elements to reflect the characteristics and communication of the product. After you have selected the brand color, we will automatically generate a complete color palette and assign it effective design semantics.	`string`	□ `#1677ff`
colorPrimaryBorder	The stroke color under the main color gradient, used on the stroke of components such as Slider.	`string`	□ `#91caff`
colorSplit	Used as the color of separator, this color is the same as colorBorderSecondary but with transparency.	`string`	□ `rgba(5, 5, 5, 0.06)`
colorText	Default text color which comply with W3C standards, and this color is also the darkest neutral color.	`string`	□ `rgba(0, 0, 0, 0.88)`
colorTextDescription	Control the font color of text description.	`string`	□ `rgba(0, 0, 0, 0.45)`
colorTextDisabled	Control the color of text in disabled state.	`string`	□ `rgba(0, 0, 0, 0.25)`
colorTextLightSolid	Control the highlight color of text with background color, such as the text in Primary Button components.	`string`	□ `#fff`
borderRadius	Border radius of base components	`number`	6
borderRadiusLG	LG size border radius, used in some large border radius components, such as Card, Modal and other components.	`number`	8
borderRadiusSM	SM size border radius, used in small size components, such as Button, Input, Select and other input components in small size	`number`	4
boxShadowSecondary	Control the secondary box shadow style of an element.	`string`	0 6px 16px 0 rgba(0, 0, 0, 0.08), 0 3px 6px -4px rgba(0, 0, 0, 0.12), 0 9px 28px 8px rgba(0, 0, 0, 0.05)
controlHeightLG	LG component height	`number`	40
controlHeightSM	SM component height	`number`	24
controlItemBgActive	Control the background color of control component item when active.	`string`	□ `#e6f4ff`
fontFamily	The font family of Ant Design prioritizes the default interface font of the system, and provides a set of alternative font libraries that are suitable for screen display to	`string`	-apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, 'Helvetica Neue', Arial, 'Noto Sans', sans-serif, 'Apple

Token Name	Description	Type	Default Value
	maintain the readability and readability of the font under different platforms and browsers, reflecting the friendly, stable and professional characteristics.		Color Emoji', 'Segoe UI Emoji', 'Segoe UI Symbol', 'Noto Color Emoji'
fontSize	The most widely used font size in the design system, from which the text gradient will be derived.	`number`	14
fontSizeLG	Large font size	`number`	16
fontSizeSM	Small font size	`number`	12
lineHeight	Line height of text.	`number`	1.5714285714285714
lineType	Border style of base components	`string`	solid
lineWidth	Border width of base components	`number`	1
lineWidthBold	The default line width of the outline class components, such as Button, Input, Select, etc.	`number`	2
lineWidthFocus	Control the width of the line when the component is in focus state.	`number`	4
margin	Control the margin of an element, with a medium size.	`number`	16
marginXS	Control the margin of an element, with a small size.	`number`	8
marginXXS	Control the margin of an element, with the smallest size.	`number`	4
motionDurationMid	Motion speed, medium speed. Used for medium element animation interaction.	`string`	0.2s
motionDurationSlow	Motion speed, slow speed. Used for large element animation interaction.	`string`	0.3s
motionEaseInOut	Preset motion curve.	`string`	cubic-bezier(0.645, 0.045, 0.355, 1)
motionEaseInOutCirc	Preset motion curve.	`string`	cubic-bezier(0.78, 0.14, 0.15, 0.86)
motionEaseInQuint	Preset motion curve.	`string`	cubic-bezier(0.755, 0.05, 0.855, 0.06)
motionEaseOut	Preset motion curve.	`string`	cubic-bezier(0.215, 0.61, 0.355, 1)
motionEaseOutCirc	Preset motion curve.	`string`	cubic-bezier(0.08, 0.82, 0.17, 1)
motionEaseOutQuint	Preset motion curve.	`string`	cubic-bezier(0.23, 1, 0.32, 1)
padding	Control the padding of the element.	`number`	16

Token Name	Description	Type	Default Value
paddingXL	Control the extra large padding of the element.	`number`	32
paddingXS	Control the extra small padding of the element.	`number`	8
zIndexPopupBase	Base zIndex of component like FloatButton, Affix which can be cover by large popup	`number`	1000