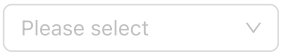
Cascader

Cascade selection box.

When To Use

- When you need to select from a set of associated data set. Such as province/city/district, company level, things classification.
- When selecting from a large data set, with multi-stage classification separated for easy selection.
- Chooses cascade items in one float layer for better user experience.

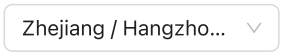
Examples



Basic

Cascade selection box for selecting province/city/district.

```
import React from 'react';
import { Cascader } from 'antd';

interface Option {
  value: string | number;
  label: string;
  children?: Option[];
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const onChange = (value: string[]) => {
  console.log(value);
};

const App: React.FC = () => (
  <Cascader options={options} onChange={onChar
);

export default App;
```



Default value

Specifies default value by an array.

```
import React from 'react';
import { Cascader } from 'antd';

interface Option {
  value: string;
  label: string;
  children?: Option[];
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const onChange = (value: string[]) => {
  console.log(value);
};

const App: React.FC = () => (
  <Cascader defaultValue={['zhejiang', 'hangzh
);

export default App;
```

Unselect Change city

Custom trigger

Separate trigger button and result.

```
import React, { useState } from 'react';
import { Cascader } from 'antd';

interface Option {
  value: string;
  label: string;
  children?: Option[];
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
      },
    ],
  },
];

const App: React.FC = () => {
  const [text, setText] = useState('Unselect')

  const onChange = (_: string[], selectedOptic
    setText(selectedOptions.map((o) => o.label
  };

  return (
    <span>
      {text}
       
      <Cascader options={options} onChange={or
        <a href="#">Change city</a>
      </Cascader>
    </span>
  );
};

export default App;
```

Hover

Hover to expand sub menu, click to select option.

```
import React from 'react';
import { Cascader } from 'antd';

interface Option {
  value: string;
  label: string;
  children?: Option[];
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const onChange = (value: string[]) => {
  console.log(value);
};

// Just show the latest item.
const displayRender = (labels: string[]) => lc

const App: React.FC = () => (
  <Cascader
    options={options}
    expandTrigger="hover"
    displayRender={displayRender}
    onChange={onChange}
  />
);

export default App;
```

Disabled option

Disable option by specifying the `disabled` property in `options`.

```
import React from 'react';
import { Cascader } from 'antd';

interface Option {
  value: string;
  label: string;
  disabled?: boolean;
  children?: Option[];
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    disabled: true,
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const onChange = (value: string[]) => {
  console.log(value);
};

const App: React.FC = () => <Cascader options=

export default App;
```


Change on select

Allow only select parent options.

```
import React from 'react';
import { Cascader } from 'antd';

interface Option {
  value: string;
  label: string;
  children?: Option[];
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hanzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const onChange = (value: string[]) => {
  console.log(value);
};

const App: React.FC = () => <Cascader options=

export default App;
```

Multiple

Select multiple options

```
import React from 'react';
import { Cascader } from 'antd';

interface Option {
  value: string | number;
  label: string;
  children?: Option[];
}

const options: Option[] = [
  {
    label: 'Light',
    value: 'light',
    children: new Array(20)
      .fill(null)
      .map((_, index) => ({ label: `Number ${i
  },
  {
    label: 'Bamboo',
    value: 'bamboo',
    children: [
      {
        label: 'Little',
        value: 'little',
        children: [
          {
            label: 'Toy Fish',
            value: 'fish',
          },
          {
            label: 'Toy Cards',
            value: 'cards',
          },
          {
            label: 'Toy Bird',
            value: 'bird',
          },
        ],
      },
    ],
  },
];

const onChange = (value: string[][]) => {
  console.log(value);
};

const App: React.FC = () => (
  <Cascader
    style={{ width: '100%' }}
    options={options}
    onChange={onChange}
    multiple
    maxTagCount="responsive"
  />
);

export default App;
```


ShowCheckedStrategy

The way show selected item in box using `ShowCheckedStrategy`.

```
import React from 'react';
import { Cascader } from 'antd';

const { SHOW_CHILD } = Cascader;

interface Option {
  value: string | number;
  label: string;
  children?: Option[];
}
const options: Option[] = [
  {
    label: 'Light',
    value: 'light',
    children: new Array(20)
      .fill(null)
      .map((_, index) => ({ label: `Number ${i
  },
  {
    label: 'Bamboo',
    value: 'bamboo',
    children: [
      {
        label: 'Little',
        value: 'little',
        children: [
          {
            label: 'Toy Fish',
            value: 'fish',
          },
          {
            label: 'Toy Cards',
            value: 'cards',
          },
          {
            label: 'Toy Bird',
            value: 'bird',
          },
        ],
      },
    ],
  },
];

const App: React.FC = () => {
  const onChange = (value: string[][]) => {
    console.log(value);
  };
  return (
    <>
      <Cascader
        style={{ width: '100%' }}
        options={options}
        onChange={onChange}
        multiple
        maxTagCount="responsive"
        showCheckedStrategy={SHOW_CHILD}
        defaultValue={[
          ['bamboo', 'little', 'fish'],
          ['bamboo', 'little', 'cards'],
```

Size

Cascade selection box of different sizes.

```
import React from 'react';
import { Cascader } from 'antd';

interface Option {
  value: string;
  label: string;
  children?: Option[];
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const onChange = (value: string[]) => {
  console.log(value);
};

const App: React.FC = () => (
  <>
    <Cascader size="large" options={options} c
    <br />
    <br />
    <Cascader options={options} onChange={onCh
    <br />
    <br />
    <Cascader size="small" options={options} c
```

Custom render

For instance, add an external link after the selected value.

```
import React from 'react';
import { Cascader } from 'antd';
import type { DefaultOptionType } from 'antd/e

interface Option {
  value: string;
  label: string;
  children?: Option[];
  code?: number;
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
            code: 752100,
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua Men',
            code: 453400,
          },
        ],
      },
    ],
  },
];

const handleAreaClick = (
  e: React.MouseEvent<HTMLAnchorElement>,
  label: string,
  option: DefaultOptionType,
) => {
  e.stopPropagation();
  console.log('clicked', label, option);
};

const displayRender = (labels: string[], selec
  labels.map((label, i) => {
    const option = selectedOptions[i];
    if (i === labels.length - 1) {
      return (
        <span key={option.value}>
```

Please select

Search

Search and select options directly.

> Now, `Cascader[showSearch]` doesn't support search on server, more info #5547

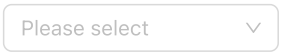
```
import React from 'react';
import { Cascader } from 'antd';
import type { DefaultOptionType } from 'antd/e

interface Option {
  value: string;
  label: string;
  children?: Option[];
  disabled?: boolean;
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
          {
            value: 'xiasha',
            label: 'Xia Sha',
            disabled: true,
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua men',
          },
        ],
      },
    ],
  },
];

const onChange = (value: string[], selectedOpt
  console.log(value, selectedOptions);
};

const filter = (inputValue: string, path: Defa
  path.some(
    (option) => (option.label as string).toLov
  );

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

Load Options Lazily

Load options lazily with `loadData`.

> Note: `loadData` cannot work with `showSearch`.

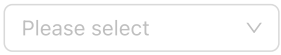
```
import React, { useState } from 'react';
import { Cascader } from 'antd';

interface Option {
  value?: string | number | null;
  label: React.ReactNode;
  children?: Option[];
  isLeaf?: boolean;
  loading?: boolean;
}

const optionLists: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    isLeaf: false,
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    isLeaf: false,
  },
];

const App: React.FC = () => {
  const [options, setOptions] = useState<Optic

  const onChange = (value: (string | number)[
    console.log(value, selectedOptions);
  };

  const loadData = (selectedOptions: Option[])
    const targetOption = selectedOptions[selec
    targetOption.loading = true;

    // load options lazily
    setTimeout(() => {
      targetOption.loading = false;
      targetOption.children = [
        {
          label: `${targetOption.label} Dynami
          value: 'dynamic1',
        },
        {
          label: `${targetOption.label} Dynami
          value: 'dynamic2',
        },
      ];
      setOptions([...options]);
    }, 1000);
  };

  return <Cascader options={options} loadData=
};

export default App;
```



Custom Field Names

Custom field names.

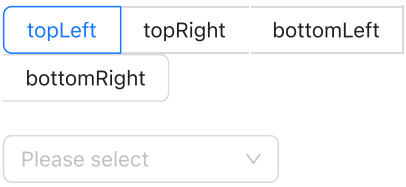
```
import React from 'react';
import { Cascader } from 'antd';

interface Option {
  code: string;
  name: string;
  items?: Option[];
}

const options: Option[] = [
  {
    code: 'zhejiang',
    name: 'Zhejiang',
    items: [
      {
        code: 'hangzhou',
        name: 'Hangzhou',
        items: [
          {
            code: 'xihu',
            name: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    code: 'jiangsu',
    name: 'Jiangsu',
    items: [
      {
        code: 'nanjing',
        name: 'Nanjing',
        items: [
          {
            code: 'zhonghuamen',
            name: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const onChange = (value: string[]) => {
  console.log(value);
};

const App: React.FC = () => (
  <Cascader
    fieldNames={{ label: 'name', value: 'code'
    options={options}
    onChange={onChange}
    placeholder="Please select"
  />
);

export default App;
```



Custom dropdown

Customize the dropdown menu via `dropdownRender`.

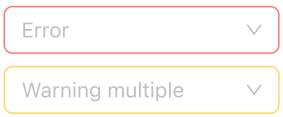
```
import React from 'react';
import { Cascader, Divider } from 'antd';

interface Option {
  value: string;
  label: string;
  children?: Option[];
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const dropdownRender = (menus: React.ReactNode
  <div>
    {menus}
    <Divider style={{ margin: 0 }} />
    <div style={{ padding: 8 }}>The footer is
  </div>
);

const App: React.FC = () => (
  <Cascader options={options} dropdownRender={
);

export default App;
```



Placement

You can manually specify the position of the popup via `placement`.

```
import React, { useState } from 'react';
import type { RadioChangeEvent } from 'antd';
import { Cascader, Radio } from 'antd';

interface Option {
  value: string;
  label: string;
  children?: Option[];
}

const options: Option[] = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
            label: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const App: React.FC = () => {
  const [placement, SetPlacement] = useState<'
    'topLeft',
  );

  const placementChange = (e: RadioChangeEvent
    SetPlacement(e.target.value);
  };

  return (
    <>
      <Radio.Group value={placement} onChange=
        <Radio.Button value="topLeft">topLeft<
```



Status

Add status to Cascader with `status`, which could be `error` or `warning`.

```
import React from 'react';
import { Cascader, Space } from 'antd';

const App: React.FC = () => (
  <Space direction="vertical">
    <Cascader status="error" placeholder="Erro
    <Cascader status="warning" multiple placeh
  </Space>
);

export default App;
```

```
      <Radio.Button value="topRight">topRigh
      <Radio.Button value="bottomLeft">bottc
      <Radio.Button value="bottomRight">bott
    </Radio.Group>
    <br />
<Cascader options={options} onChange={onChange} />
  </>
);
export default App;
```

Property	Description	Type	Default
allowClear	Whether allow clear	boolean	true
autoFocus	If get focus when component mounted	boolean	false
bordered	Whether has border style	boolean	true
clearIcon	The custom clear icon	ReactNode	-
changeOnSelect	(Work on single select) Change value on each selection if set to true, see above demo for details	boolean	false
className	The additional css class	string	-
defaultValue	Initial selected value	string[] | number[]	[]
disabled	Whether disabled select	boolean	false
displayRender	The render function of displaying selected options	(label, selectedOptions) => ReactNode	label => label.joi
popupClassName	The additional className of popup overlay	string	-
dropdownRender	Customize dropdown content	(menus: ReactNode) => ReactNode	-
expandIcon	Customize the current item expand icon	ReactNode	-
expandTrigger	expand current item when click or hover, one of `click` `hover`	string	`click`
fieldNames	Custom field name for label and value and children	object	{ label: value: `va` children: }
getPopupContainer	Parent Node which the selector should be rendered to. Default to `body`. When position issues	function(triggerNode)	() => doc

Property	Description	Type	Default
	happen, try to modify it into scrollable content and position it relative. example		
loadData	To load option lazily, and it cannot work with `showSearch`	`(selectedOptions) => void`	-
maxTagCount	Max tag count to show. `responsive` will cost render performance	number | `responsive`	-
maxTagPlaceholder	Placeholder for not showing tags	ReactNode | function(omittedValues)	-
maxTagTextLength	Max tag text length to show	number	-
notFoundContent	Specify content to show when no result matches	string	`Not Found`
open	Set visible of cascader popup	boolean	-
options	The data options of cascade	Option[]	-
placeholder	The input placeholder	string	`Please sel`
placement	Use preset popup align config from builtinPlacements	`bottomLeft` `bottomRight` `topLeft` `topRight`	`bottomLeft`
showSearch	Whether show search input in single mode	boolean | Object	false
size	The input size	`large` | `middle` | `small`	-
status	Set validation status	'error' | 'warning'	-
style	The additional style	CSSProperties	-
suffixIcon	The custom suffix icon	ReactNode	-
value	The selected value	string[] | number[]	-
onChange	Callback when finishing cascader select	`(value, selectedOptions) => void`	-
onDropdownVisibleChange	Callback when popup shown or hidden	`(value) => void`	-
multiple	Support multiple or	boolean	-

Property	Description	Type	Default
	not		
removeIcon	The custom remove icon	ReactNode	-
showCheckedStrategy	The way show selected item in box. ** `SHOW_CHILD` : ** just show child treeNode. `Cascader.SHOW_PARENT` : just show parent treeNode (when all child treeNode under the parent treeNode are checked)	`Cascader.SHOW_PARENT` | `Cascader.SHOW_CHILD`	`Cascader.S`
searchValue	Set search value, Need work with `showSearch`	string	-
onSearch	The callback function triggered when input changed	(search: string) => void	-
dropdownMenuColumnStyle	The style of the drop-down menu column	CSSProperties	-
loadingIcon	The appearance of lazy loading (now is useless)	ReactNode	-

showSearch

Property	Description	Type	Default	Version
filter	The function will receive two arguments, inputValue and option, if the function returns true, the option will be included in the filtered set; Otherwise, it will be excluded	function(inputValue, path): boolean	-	
limit	Set the count of filtered items	number | false	50	
matchInputWidth	Whether the width of list matches input, (how it looks)	boolean	true	
render	Used to render filtered options	function(inputValue, path): ReactNode	-	
sort	Used to sort filtered options	function(a, b, inputValue)	-	

Option

```ts
interface Option {
  value: string | number;
  label?: React.ReactNode;
  disabled?: boolean;
  children?: Option[];
  // Determines if this is a leaf node(effective when `loadData` is specified).
  // `false` will force trade TreeNode as a parent node.
  // Show expand icon even if the current node has no children.
  isLeaf?: boolean;
}
```

Methods

Name	Description	Version
blur()	Remove focus	
focus()	Get focus	

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
colorBgContainerDisabled	Control the background color of container in disabled state.	string	rgba(0, 0, 0, 0.04)
colorBorder	Default border color, used to separate different elements, such as: form separator, card separator, etc.	string	#d9d9d9
colorHighlight	Control the color of page element when highlighted.	string	#ff4d4f
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
colorPrimaryHover	Hover state under the main color gradient.	string	#4096ff
colorSplit	Used as the color of separator, this color is the same as colorBorderSecondary but	string	rgba(5, 5, 5, 0.06)

Token Name	Description	Type	Default Value
	with transparency.		
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)`
colorWhite	Pure white color don't changed by theme	`string`	`#fff`
borderRadiusSM	SM size border radius, used in small size components, such as Button, Input, Select and other input components in small size	`number`	4
controlHeight	The height of the basic controls such as buttons and input boxes in Ant Design	`number`	32
controlInteractiveSize	Control the interactive size of control component.	`number`	16
controlItemBgActive	Control the background color of control component item when active.	`string`	`#e6f4ff`
controlItemBgHover	Control the background color of control component item when hovering.	`string`	`rgba(0, 0, 0, 0.04)`
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 maintain the readability and readability of the font under different platforms and browsers, reflecting the friendly, stable and professional characteristics.	`string`	-apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, 'Helvetica Neue', Arial, 'Noto Sans', sans-serif, 'Apple 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
fontSizeIcon	Control the font size of operation icon in Select, Cascader, etc. Normally same as fontSizeSM.	`number`	12
fontSizeLG	Large font size	`number`	16
fontWeightStrong	Control the font weight of heading components (such as h1, h2, h3) or selected item.	`number`	600
lineHeight	Line height of text.	`number`	1.5714285714285714
lineType	Border style of base components	`string`	solid
lineWidth	Border width of base components	`number`	1

Token Name	Description	Type	Default Value
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
marginXS	Control the margin of an element, with a small size.	`number`	8
motionDurationFast	Motion speed, fast speed. Used for small element animation interaction.	`string`	0.1s
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
motionEaseInBack	Preset motion curve.	`string`	cubic-bezier(0.71, -0.46, 0.88, 0.6)
motionEaseOutBack	Preset motion curve.	`string`	cubic-bezier(0.12, 0.4, 0.29, 1.46)
paddingSM	Control the small padding of the element.	`number`	12
paddingXS	Control the extra small padding of the element.	`number`	8
paddingXXS	Control the extra extra small padding of the element.	`number`	4