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 classifications separated for easy selection.
- Chooses cascade items in one float layer for better user experience.

Examples

Basic

```tsx
import React from 'react';
import type { CascaderProps } from 'antd';
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: CascaderProps<Option>['onChange'] = (value) => {
  console.log(value);
};

const App: React.FC = () => (
  <Cascader options={options} onChange={onChange} placeholder="Please
select" />
);

export default App;
```

Default value

```
import React from 'react';
import type { CascaderProps } from 'antd';
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: CascaderProps<Option>['onChange'] = (value) => {
  console.log(value);
};

const App: React.FC = () => (
  <Cascader defaultValue={['zhejiang', 'hangzhou', 'xihu']} options=
{options} onChange={onChange} />
);

export default App;
```

Custom trigger

```
import React, { useState } from 'react';
import type { CascaderProps } from 'antd';
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: CascaderProps<Option>['onChange'] = (_, selectedOptions)
=> {
    setText(selectedOptions.map((o) => o.label).join(', '));
  };

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

export default App;
```

Hover

```tsx
import React from 'react';
import type { CascaderProps } from 'antd';
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: CascaderProps<Option>['onChange'] = (value) => {
```

```
  console.log(value);
};

// Just show the latest item.
const displayRender = (labels: string[]) => labels[labels.length - 1];

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

export default App;
```

Disabled option

```
import React from 'react';
import type { CascaderProps } from 'antd';
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: CascaderProps<Option>['onChange'] = (value) => {
  console.log(value);
};

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

export default App;
```

Change on select

```
import React from 'react';
import type { CascaderProps } from 'antd';
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: CascaderProps<Option>['onChange'] = (value) => {
  console.log(value);
};

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

export default App;
```

Multiple

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

interface Option {
```

```tsx
  value: string | number;
  label: string;
  children?: Option[];
  disableCheckbox?: boolean;
}

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

const onChange: CascaderProps<Option, 'value', true>['onChange'] = (value)
=> {
  console.log(value);
};
```

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

export default App;
```

ShowCheckedStrategy

```tsx
import React from 'react';
import type { CascaderProps } from 'antd';
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: Array.from({ length: 20 }).map((_, index) => ({
      label: `Number ${index}`,
      value: index,
    })),
  },
  {
    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: CascaderProps<Option, 'value', true>['onChange'] =
(value) => {
    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'],
          ['bamboo', 'little', 'bird'],
        ]}
      />
      <br />
      <br />
      <Cascader
        style={{ width: '100%' }}
        options={options}
        onChange={onChange}
        multiple
        maxTagCount="responsive"
        defaultValue={[['bamboo']]}
      />
    </>
```

```
  );
};

export default App;
```

Size

```
import React from 'react';
import type { CascaderProps } from 'antd';
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: CascaderProps<Option>['onChange'] = (value) => {
  console.log(value);
};

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

export default App;
```

Custom render

```
import React from 'react';
import { Cascader } from 'antd';
import type { CascaderProps, GetProp } from 'antd';

type DefaultOptionType = GetProp<CascaderProps, 'options'>[number];

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: CascaderProps<Option>['displayRender'] = (labels,
selectedOptions = []) =>
  labels.map((label, i) => {
    const option = selectedOptions[i];
    if (i === labels.length - 1) {
```

```
      return (
        <span key={option.value}>
          {label} (<a onClick={(e) => handleAreaClick(e, label, option)}>
{option.code}</a>)
        </span>
      );
    }
    return <span key={option.value}>{label} / </span>;
  });

const App: React.FC = () => (
  <Cascader
    options={options}
    defaultValue={['zhejiang', 'hangzhou', 'xihu']}
    displayRender={displayRender}
    style={{ width: '100%' }}
  />
);

export default App;
```

Search

```
import React from 'react';
import { Cascader } from 'antd';
import type { CascaderProps, GetProp } from 'antd';

type DefaultOptionType = GetProp<CascaderProps, 'options'>[number];

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: CascaderProps<Option>['onChange'] = (value,
selectedOptions) => {
  console.log(value, selectedOptions);
};

const filter = (inputValue: string, path: DefaultOptionType[]) =>
  path.some(
    (option) => (option.label as
string).toLowerCase().indexOf(inputValue.toLowerCase()) > -1,
  );

const App: React.FC = () => (
  <Cascader
    options={options}
    onChange={onChange}
    placeholder="Please select"
```

```
    showSearch={{ filter }}
    onSearch={(value) => console.log(value)}
  />
);

export default App;
```

Load Options Lazily

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

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

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

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

  const onChange: CascaderProps<Option>['onChange'] = (value,
selectedOptions) => {
    console.log(value, selectedOptions);
  };

  const loadData = (selectedOptions: Option[]) => {
    const targetOption = selectedOptions[selectedOptions.length - 1];

    // load options lazily
    setTimeout(() => {
```

```
      targetOption.children = [
        {
          label: `${targetOption.label} Dynamic 1`,
          value: 'dynamic1',
        },
        {
          label: `${targetOption.label} Dynamic 2`,
          value: 'dynamic2',
        },
      ];
      setOptions([...options]);
    }, 1000);
  };

  return <Cascader options={options} loadData={loadData} onChange=
{onChange} changeOnSelect />;
};

export default App;
```

Custom Field Names

```
import React from 'react';
import type { CascaderProps } from 'antd';
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: CascaderProps<Option>['onChange'] = (value) => {
  console.log(value);
};

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

export default App;
```

Prefix and Suffix

v5.22.0

```
import React from 'react';
import { SmileOutlined } from '@ant-design/icons';
import type { CascaderProps } from 'antd';
import { Cascader } from 'antd';

interface Option {
```

```tsx
  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: CascaderProps<Option>['onChange'] = (value) => {
  console.log(value);
};

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

```
    <Cascader
      suffixIcon={<SmileOutlined />}
      options={options}
      onChange={onChange}
      placeholder="Please select"
    />
    <br />
    <br />
    <Cascader suffixIcon="ab" options={options} onChange={onChange}
placeholder="Please select" />
    <br />
    <br />
    <Cascader
      expandIcon={<SmileOutlined />}
      options={options}
      onChange={onChange}
      placeholder="Please select"
    />
    <br />
    <br />
    <Cascader expandIcon="ab" options={options} onChange={onChange}
placeholder="Please select" />
    <br />
    <br />
    <Cascader
      prefix={<SmileOutlined />}
      options={options}
      onChange={onChange}
      placeholder="Please select"
    />
  </>
);

export default App;
```

Custom dropdown

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

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

```tsx
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 not very short.</div>
  </div>
);

const App: React.FC = () => (
  <Cascader options={options} dropdownRender={dropdownRender}
placeholder="Please select" />
```

```
);

export default App;
```

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<'bottomLeft' | 'bottomRight' |
'topLeft' | 'topRight'>(
    'topLeft',
  );

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

  return (
    <>
      <Radio.Group value={placement} onChange={placementChange}>
        <Radio.Button value="topLeft">topLeft</Radio.Button>
        <Radio.Button value="topRight">topRight</Radio.Button>
        <Radio.Button value="bottomLeft">bottomLeft</Radio.Button>
        <Radio.Button value="bottomRight">bottomRight</Radio.Button>
      </Radio.Group>
      <br />
      <br />
      <Cascader options={options} placeholder="Please select" placement=
{placement} />
    </>
  );
};

export default App;
```

Variants

v5.13.0

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

const App: React.FC = () => (
  <Flex vertical gap="middle">
    <Cascader placeholder="Please select" variant="borderless" />
    <Cascader placeholder="Please select" variant="filled" />
    <Cascader placeholder="Please select" variant="outlined" />
    <Cascader placeholder="Please select" variant="underlined" />
```

```
  </Flex>
);

export default App;
```

Status

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

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

export default App;
```

= 5.10.0">Panel

```
import React, { useState } from 'react';
import type { CascaderProps } from 'antd';
import { Cascader, Flex, Switch } 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: CascaderProps<Option>['onChange'] = (value) => {
  console.log(value);
};

const onMultipleChange: CascaderProps<Option, 'value', true>['onChange'] =
(value) => {
  console.log(value);
};

const App: React.FC = () => {
  const [disabled, setDisabled] = useState(false);

  return (
    <Flex vertical gap="small" align="flex-start">
      <Switch
        checked={disabled}
        checkedChildren="Enabled"
        unCheckedChildren="Disabled"
        onChange={setDisabled}
        aria-label="disabled switch"
      />
      <Cascader.Panel options={options} onChange={onChange} disabled=
{disabled} />
      <Cascader.Panel multiple options={options} onChange=
{onMultipleChange} disabled={disabled} />
```

```
      <Cascader.Panel />
    </Flex>
  );
};

export default App;
```

_InternalPanelDoNotUseOrYouWillBeFired

Debug

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

const { _InternalPanelDoNotUseOrYouWillBeFired: InternalCascader } =
Cascader;

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 App: React.FC = () => <InternalCascader options={options}
placeholder="Please select" />;

export default App;
```

Component Token

Debug

```
import React from 'react';
import type { CascaderProps } from 'antd';
import { Cascader, ConfigProvider } 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: CascaderProps<Option>['onChange'] = (value) => {
  console.log(value);
};

const App: React.FC = () => (
  <ConfigProvider
    theme={{
      components: {
        Cascader: {
          optionSelectedColor: 'red',
        },
      },
    }}
  >
    <Cascader options={options} onChange={onChange} placeholder="Please
select" />
  </ConfigProvider>
);

export default App;
```

API

Common props ref：Common props

```
<Cascader options={options} onChange={onChange} />
```

Property	Description	Type	
allowClear	Show clear button	boolean | { clearIcon?: ReactNode }	true
autoClearSearchValue	Whether the current search will be cleared on selecting an item. Only applies when `multiple` is `true`	boolean	true
autoFocus	If get focus when component mounted	boolean	false
changeOnSelect	Change value on each selection if set to true, see above demo for details	boolean	false
className	The additional css class	string	-
defaultOpen	Initial visible of cascader popup	boolean	-
defaultValue	Initial selected value	string[] | number[]	[]
disabled	Whether disabled select	boolean	false
displayRender	The render function of displaying selected options	(label, selectedOptions) => ReactNode	label =
tagRender	Custom render function for tags in `multiple` mode	(label: string, onClose: function, value: string) => ReactNode	-
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	string	`click`

	of `click` `hover`		

fieldNames	Custom field name for label and value and children	object	{ label
			`value`
			`child`

getPopupContainer	Parent Node which the selector should be rendered to. Default to `body`. When position issues happen, try to modify it into scrollable content and position it relative. example	function(triggerNode)	() =>
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	ReactNode	`Not F`
open	Set visible of cascader popup	boolean	-
options	The data options of cascade	Option[]	-
placeholder	The input placeholder	string	-
placement	Use preset popup align config from builtinPlacements	`bottomLeft` `bottomRight` `topLeft` `topRight`	`botto`
prefix	The custom prefix	ReactNode	-
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[]	-
variant	Variants of selector	`outlined` | `borderless` | `filled` | `underlined`	`outli`
onChange	Callback when finishing cascader select	(value, selectedOptions) => void	-
onDropdownVisibleChange	Callback when popup shown or hidden	(value) => void	-
multiple	Support multiple or not	boolean	-
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`	`Casca`
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	-
optionRender	Customize the rendering dropdown options	(option: Option) => React.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

```
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