When To Use

- A dropdown menu for displaying choices - an elegant alternative to the native `<select>` element.
- Utilizing Radio is recommended when there are fewer total options (less than 5).
- You probably need AutoComplete if you're looking for an input box that can be typed or selected.

Usage upgrade 5.11.0+

:::warning{title="Upgrade Tip"} After version 5.11.0, we provide a simpler usage `<Select options={[...]} />` with better performance and potential of writing simpler code style in your applications. Meanwhile, we deprecated the old usage in browser console, we will remove it in antd 6.0. :::

```
// works when >=5.11.0, recommended ✅
return <Select options={[{ value: 'sample', label: <span>sample</span> }]}
/>;

// works when <5.11.0, deprecated when >=5.11.0 🧕
return (
  <Select onChange={onChange}>
    <Select.Option value="sample">Sample</Select.Option>
  </Select>
);
```

Examples

Basic Usage

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

const handleChange = (value: string) => {
  console.log(`selected ${value}`);
};

const App: React.FC = () => (
  <Space wrap>
    <Select
      defaultValue="lucy"
      style={{ width: 120 }}
      onChange={handleChange}
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'Yiminghe', label: 'yiminghe' },
```

```
        { value: 'disabled', label: 'Disabled', disabled: true },
      ]}
    />
    <Select
      defaultValue="lucy"
      style={{ width: 120 }}
      disabled
      options={[{ value: 'lucy', label: 'Lucy' }]}
    />
    <Select
      defaultValue="lucy"
      style={{ width: 120 }}
      loading
      options={[{ value: 'lucy', label: 'Lucy' }]}
    />
    <Select
      defaultValue="lucy"
      style={{ width: 120 }}
      allowClear
      options={[{ value: 'lucy', label: 'Lucy' }]}
      placeholder="select it"
    />
  </Space>
);

export default App;
```

Select with search field

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

const onChange = (value: string) => {
  console.log(`selected ${value}`);
};

const onSearch = (value: string) => {
  console.log('search:', value);
};

const App: React.FC = () => (
  <Select
    showSearch
    placeholder="Select a person"
    optionFilterProp="label"
    onChange={onChange}
```

```
    onSearch={onSearch}
    options={[
      {
        value: 'jack',
        label: 'Jack',
      },
      {
        value: 'lucy',
        label: 'Lucy',
      },
      {
        value: 'tom',
        label: 'Tom',
      },
    ]}
  />
);

export default App;
```

Custom Search

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

const App: React.FC = () => (
  <Select
    showSearch
    placeholder="Select a person"
    filterOption={(input, option) =>
      (option?.label ?? '').toLowerCase().includes(input.toLowerCase())
    }
    options={[
      { value: '1', label: 'Jack' },
      { value: '2', label: 'Lucy' },
      { value: '3', label: 'Tom' },
    ]}
  />
);

export default App;
```

multiple selection

```tsx
import React from 'react';
import { Select, Space } from 'antd';
import type { SelectProps } from 'antd';

const options: SelectProps['options'] = [];

for (let i = 10; i < 36; i++) {
  options.push({
    label: i.toString(36) + i,
    value: i.toString(36) + i,
  });
}

const handleChange = (value: string[]) => {
  console.log(`selected ${value}`);
};

const App: React.FC = () => (
  <Space style={{ width: '100%' }} direction="vertical">
    <Select
      mode="multiple"
      allowClear
      style={{ width: '100%' }}
      placeholder="Please select"
      defaultValue={['a10', 'c12']}
      onChange={handleChange}
      options={options}
    />
    <Select
      mode="multiple"
      disabled
      style={{ width: '100%' }}
      placeholder="Please select"
      defaultValue={['a10', 'c12']}
      onChange={handleChange}
      options={options}
    />
  </Space>
);

export default App;
```

Sizes

```tsx
import React, { useState } from 'react';
import { Radio, Select, Space } from 'antd';
import type { ConfigProviderProps, RadioChangeEvent, SelectProps } from
'antd';

type SizeType = ConfigProviderProps['componentSize'];

const options: SelectProps['options'] = [];

for (let i = 10; i < 36; i++) {
  options.push({
    value: i.toString(36) + i,
    label: i.toString(36) + i,
  });
}

const handleChange = (value: string | string[]) => {
  console.log(`Selected: ${value}`);
};

const App: React.FC = () => {
  const [size, setSize] = useState<SizeType>('middle');

  const handleSizeChange = (e: RadioChangeEvent) => {
    setSize(e.target.value);
  };

  return (
    <>
      <Radio.Group value={size} onChange={handleSizeChange}>
        <Radio.Button value="large">Large</Radio.Button>
        <Radio.Button value="middle">Default</Radio.Button>
        <Radio.Button value="small">Small</Radio.Button>
      </Radio.Group>
      <br />
      <br />
      <Space direction="vertical" style={{ width: '100%' }}>
        <Select
          size={size}
          defaultValue="a1"
          onChange={handleChange}
          style={{ width: 200 }}
          options={options}
        />
        <Select
          mode="multiple"
```

```
          size={size}
          placeholder="Please select"
          defaultValue={['a10', 'c12']}
          onChange={handleChange}
          style={{ width: '100%' }}
          options={options}
        />
        <Select
          mode="tags"
          size={size}
          placeholder="Please select"
          defaultValue={['a10', 'c12']}
          onChange={handleChange}
          style={{ width: '100%' }}
          options={options}
        />
      </Space>
    </>
  );
};

export default App;
```

Custom dropdown options

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

const handleChange = (value: string[]) => {
  console.log(`selected ${value}`);
};

const options = [
  {
    label: 'China',
    value: 'china',
    emoji: '🇨🇳',
    desc: 'China (中国)',
  },
  {
    label: 'USA',
    value: 'usa',
    emoji: '🇺🇸',
    desc: 'USA (美国)',
  },
  {
```

```
    label: 'Japan',
    value: 'japan',
    emoji: '🇯🇵',
    desc: 'Japan (日本)',
  },
  {
    label: 'Korea',
    value: 'korea',
    emoji: '🇰🇷',
    desc: 'Korea (韩国)',
  },
];

const App: React.FC = () => (
  <Select
    mode="multiple"
    style={{ width: '100%' }}
    placeholder="select one country"
    defaultValue={['china']}
    onChange={handleChange}
    options={options}
    optionRender={(option) => (
      <Space>
        <span role="img" aria-label={option.data.label}>
          {option.data.emoji}
        </span>
        {option.data.desc}
      </Space>
    )}
  />
);

export default App;
```

Search with sort

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

const App: React.FC = () => (
  <Select
    showSearch
    style={{ width: 200 }}
    placeholder="Search to Select"
    optionFilterProp="label"
    filterSort={(optionA, optionB) =>
```

```
      (optionA?.label ?? '').toLowerCase().localeCompare((optionB?.label ??
'').toLowerCase())
    }
    options={[
      {
        value: '1',
        label: 'Not Identified',
      },
      {
        value: '2',
        label: 'Closed',
      },
      {
        value: '3',
        label: 'Communicated',
      },
      {
        value: '4',
        label: 'Identified',
      },
      {
        value: '5',
        label: 'Resolved',
      },
      {
        value: '6',
        label: 'Cancelled',
      },
    ]}
  />
);

export default App;
```

Tags

```
import React from 'react';
import { Select } from 'antd';
import type { SelectProps } from 'antd';

const options: SelectProps['options'] = [];

for (let i = 10; i < 36; i++) {
  options.push({
    value: i.toString(36) + i,
    label: i.toString(36) + i,
```

```
  });
}

const handleChange = (value: string[]) => {
  console.log(`selected ${value}`);
};

const App: React.FC = () => (
  <Select
    mode="tags"
    style={{ width: '100%' }}
    placeholder="Tags Mode"
    onChange={handleChange}
    options={options}
  />
);

export default App;
```

Option Group

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

const handleChange = (value: string) => {
  console.log(`selected ${value}`);
};

const App: React.FC = () => (
  <Select
    defaultValue="lucy"
    style={{ width: 200 }}
    onChange={handleChange}
    options={[
      {
        label: <span>manager</span>,
        title: 'manager',
        options: [
          { label: <span>Jack</span>, value: 'Jack' },
          { label: <span>Lucy</span>, value: 'Lucy' },
        ],
      },
      {
        label: <span>engineer</span>,
        title: 'engineer',
        options: [
```

```
          { label: <span>Chloe</span>, value: 'Chloe' },
          { label: <span>Lucas</span>, value: 'Lucas' },
        ],
      },
    ]}
  />
);

export default App;
```

coordinate

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

const cityData = {
  Zhejiang: ['Hangzhou', 'Ningbo', 'Wenzhou'],
  Jiangsu: ['Nanjing', 'Suzhou', 'Zhenjiang'],
};

type CityName = keyof typeof cityData;

const provinceData: CityName[] = ['Zhejiang', 'Jiangsu'];

const App: React.FC = () => {
  const [cities, setCities] = useState(cityData[provinceData[0] as
CityName]);
  const [secondCity, setSecondCity] = useState(cityData[provinceData[0]][0]
as CityName);

  const handleProvinceChange = (value: CityName) => {
    setCities(cityData[value]);
    setSecondCity(cityData[value][0] as CityName);
  };

  const onSecondCityChange = (value: CityName) => {
    setSecondCity(value);
  };

  return (
    <Space wrap>
      <Select
        defaultValue={provinceData[0]}
        style={{ width: 120 }}
        onChange={handleProvinceChange}
        options={provinceData.map((province) => ({ label: province, value:
```

```
province }))}
      />
      <Select
        style={{ width: 120 }}
        value={secondCity}
        onChange={onSecondCityChange}
        options={cities.map((city) => ({ label: city, value: city }))}
      />
    </Space>
  );
};

export default App;
```

Search Box

```
/* eslint-disable compat/compat */
import React, { useState } from 'react';
import { Select } from 'antd';
import type { SelectProps } from 'antd';
import type { AnyObject } from 'antd/es/_util/type';

let timeout: ReturnType<typeof setTimeout> | null;
let currentValue: string;

const toURLSearchParams = <T extends AnyObject>(record: T) => {
  const params = new URLSearchParams();
  for (const [key, value] of Object.entries(record)) {
    params.append(key, value);
  }
  return params;
};

const fetchData = (value: string, callback: (data: { value: string; text:
string }[]) => void) => {
  if (timeout) {
    clearTimeout(timeout);
    timeout = null;
  }
  currentValue = value;

  const params = toURLSearchParams({ code: 'utf-8', q: value });

  const fake = () => {
    fetch(`https://suggest.taobao.com/sug?${params.toString()}`)
      .then((response) => response.json())
```

```
      .then(({ result }) => {
        if (currentValue === value) {
          const data = result.map((item: any) => ({ value: item[0], text:
item[0] }));
          callback(data);
        }
      });
  };
  if (value) {
    timeout = setTimeout(fake, 300);
  } else {
    callback([]);
  }
};

const SearchInput: React.FC<{ placeholder: string; style:
React.CSSProperties }> = (props) => {
  const [data, setData] = useState<SelectProps['options']>([]);
  const [value, setValue] = useState<string>();

  const handleSearch = (newValue: string) => {
    fetchData(newValue, setData);
  };

  const handleChange = (newValue: string) => {
    setValue(newValue);
  };

  return (
    <Select
      showSearch
      value={value}
      placeholder={props.placeholder}
      style={props.style}
      defaultActiveFirstOption={false}
      suffixIcon={null}
      filterOption={false}
      onSearch={handleSearch}
      onChange={handleChange}
      notFoundContent={null}
      options={(data || []).map((d) => ({
        value: d.value,
        label: d.text,
      }))}
    />
  );
```

```
};

const App: React.FC = () => <SearchInput placeholder="input search text"
style={{ width: 200 }} />;

export default App;
```

Get value of selected item

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

const handleChange = (value: { value: string; label: React.ReactNode }) =>
{
  console.log(value); // { value: "lucy", key: "lucy", label: "Lucy (101)"
}
};

const App: React.FC = () => (
  <Select
    labelInValue
    defaultValue={{ value: 'lucy', label: 'Lucy (101)' }}
    style={{ width: 120 }}
    onChange={handleChange}
    options={[
      {
        value: 'jack',
        label: 'Jack (100)',
      },
      {
        value: 'lucy',
        label: 'Lucy (101)',
      },
    ]}
  />
);

export default App;
```

Automatic tokenization

```
import React from 'react';
import { Select } from 'antd';
import type { SelectProps } from 'antd';
```

```
const options: SelectProps['options'] = [];

for (let i = 10; i < 36; i++) {
  options.push({
    value: i.toString(36) + i,
    label: i.toString(36) + i,
  });
}

const handleChange = (value: string[]) => {
  console.log(`selected ${value}`);
};

const App: React.FC = () => (
  <Select
    mode="tags"
    style={{ width: '100%' }}
    onChange={handleChange}
    tokenSeparators={[',']}
    options={options}
  />
);

export default App;
```

Search and Select Users

```
import React, { useMemo, useRef, useState } from 'react';
import { Select, Spin } from 'antd';
import type { SelectProps } from 'antd';
import debounce from 'lodash/debounce';

export interface DebounceSelectProps<ValueType = any>
  extends Omit<SelectProps<ValueType | ValueType[]>, 'options' |
'children'> {
  fetchOptions: (search: string) => Promise<ValueType[]>;
  debounceTimeout?: number;
}

function DebounceSelect<
  ValueType extends { key?: string; label: React.ReactNode; value: string |
number } = any,
>({ fetchOptions, debounceTimeout = 800, ...props }:
DebounceSelectProps<ValueType>) {
  const [fetching, setFetching] = useState(false);
  const [options, setOptions] = useState<ValueType[]>([]);
```

```
  const fetchRef = useRef(0);

  const debounceFetcher = useMemo(() => {
    const loadOptions = (value: string) => {
      fetchRef.current += 1;
      const fetchId = fetchRef.current;
      setOptions([]);
      setFetching(true);

      fetchOptions(value).then((newOptions) => {
        if (fetchId !== fetchRef.current) {
          // for fetch callback order
          return;
        }

        setOptions(newOptions);
        setFetching(false);
      });
    };

    return debounce(loadOptions, debounceTimeout);
  }, [fetchOptions, debounceTimeout]);

  return (
    <Select
      labelInValue
      filterOption={false}
      onSearch={debounceFetcher}
      notFoundContent={fetching ? <Spin size="small" /> : null}
      {...props}
      options={options}
    />
  );
}

// Usage of DebounceSelect
interface UserValue {
  label: string;
  value: string;
}

async function fetchUserList(username: string): Promise<UserValue[]> {
  console.log('fetching user', username);

  return fetch('https://randomuser.me/api/?results=5')
    .then((response) => response.json())
```

```
    .then((body) =>
      body.results.map(
        (user: { name: { first: string; last: string }; login: { username:
string } }) => ({
          label: `${user.name.first} ${user.name.last}`,
          value: user.login.username,
        }),
      ),
    );
}

const App: React.FC = () => {
  const [value, setValue] = useState<UserValue[]>([]);

  return (
    <DebounceSelect
      mode="multiple"
      value={value}
      placeholder="Select users"
      fetchOptions={fetchUserList}
      style={{ width: '100%' }}
      onChange={(newValue) => {
        if (Array.isArray(newValue)) {
          setValue(newValue);
        }
      }}
    />
  );
};

export default App;
```

Prefix and Suffix

v5.22.0

```
import React from 'react';
import { MehOutlined, SmileOutlined } from '@ant-design/icons';
import { Select, Space } from 'antd';

const smileIcon = <SmileOutlined />;
const mehIcon = <MehOutlined />;

const handleChange = (value: string | string[]) => {
  console.log(`selected ${value}`);
};
```

```tsx
const App: React.FC = () => (
  <Space wrap>
    <Select
      prefix="User"
      defaultValue="lucy"
      style={{ width: 200 }}
      onChange={handleChange}
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'Yiminghe', label: 'yiminghe' },
        { value: 'disabled', label: 'Disabled', disabled: true },
      ]}
    />
    <Select
      suffixIcon={smileIcon}
      defaultValue="lucy"
      style={{ width: 120 }}
      onChange={handleChange}
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'Yiminghe', label: 'yiminghe' },
        { value: 'disabled', label: 'Disabled', disabled: true },
      ]}
    />
    <Select
      suffixIcon={mehIcon}
      defaultValue="lucy"
      style={{ width: 120 }}
      disabled
      options={[{ value: 'lucy', label: 'Lucy' }]}
    />
    <br />
    <Select
      prefix="User"
      defaultValue={['lucy']}
      mode="multiple"
      style={{ width: 200 }}
      onChange={handleChange}
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'Yiminghe', label: 'yiminghe' },
        { value: 'disabled', label: 'Disabled', disabled: true },
```

```
      ]}
    />
    <Select
      suffixIcon={smileIcon}
      defaultValue={['lucy']}
      mode="multiple"
      style={{ width: 120 }}
      onChange={handleChange}
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'Yiminghe', label: 'yiminghe' },
        { value: 'disabled', label: 'Disabled', disabled: true },
      ]}
    />
    <Select
      suffixIcon={mehIcon}
      defaultValue={['lucy']}
      mode="multiple"
      style={{ width: 120 }}
      disabled
      options={[{ value: 'lucy', label: 'Lucy' }]}
    />
  </Space>
);

export default App;
```

Custom dropdown

```
import React, { useRef, useState } from 'react';
import { PlusOutlined } from '@ant-design/icons';
import { Button, Divider, Input, Select, Space } from 'antd';
import type { InputRef } from 'antd';

let index = 0;

const App: React.FC = () => {
  const [items, setItems] = useState(['jack', 'lucy']);
  const [name, setName] = useState('');
  const inputRef = useRef<InputRef>(null);

  const onNameChange = (event: React.ChangeEvent<HTMLInputElement>) => {
    setName(event.target.value);
  };
```

```
  const addItem = (e: React.MouseEvent<HTMLButtonElement |
HTMLAnchorElement>) => {
    e.preventDefault();
    setItems([...items, name || `New item ${index++}`]);
    setName('');
    setTimeout(() => {
      inputRef.current?.focus();
    }, 0);
  };

  return (
    <Select
      style={{ width: 300 }}
      placeholder="custom dropdown render"
      dropdownRender={(menu) => (
        <>
          {menu}
          <Divider style={{ margin: '8px 0' }} />
          <Space style={{ padding: '0 8px 4px' }}>
            <Input
              placeholder="Please enter item"
              ref={inputRef}
              value={name}
              onChange={onNameChange}
              onKeyDown={(e) => e.stopPropagation()}
            />
            <Button type="text" icon={<PlusOutlined />} onClick={addItem}>
              Add item
            </Button>
          </Space>
        </>
      )}
      options={items.map((item) => ({ label: item, value: item }))}
    />
  );
};

export default App;
```

Hide Already Selected

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

const OPTIONS = ['Apples', 'Nails', 'Bananas', 'Helicopters'];
```

```
const App: React.FC = () => {
  const [selectedItems, setSelectedItems] = useState<string[]>([]);

  const filteredOptions = OPTIONS.filter((o) =>
!selectedItems.includes(o));

  return (
    <Select
      mode="multiple"
      placeholder="Inserted are removed"
      value={selectedItems}
      onChange={setSelectedItems}
      style={{ width: '100%' }}
      options={filteredOptions.map((item) => ({
        value: item,
        label: item,
      }))}
    />
  );
};

export default App;
```

Variants

v5.13.0

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

const App: React.FC = () => (
  <Flex gap={12} vertical>
    <Flex gap={8}>
      <Select
        placeholder="Outlined"
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
      <Select
        mode="multiple"
        defaultValue={['lucy']}
        placeholder="Outlined"
```

```
      style={{ flex: 1 }}
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'Yiminghe', label: 'yiminghe' },
      ]}
    />
  </Flex>
  <Flex gap={8}>
    <Select
      placeholder="Filled"
      variant="filled"
      style={{ flex: 1 }}
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'Yiminghe', label: 'yiminghe' },
      ]}
    />
    <Select
      mode="multiple"
      defaultValue={['lucy']}
      placeholder="Filled"
      variant="filled"
      style={{ flex: 1 }}
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'Yiminghe', label: 'yiminghe' },
      ]}
    />
  </Flex>
  <Flex gap={8}>
    <Select
      placeholder="Borderless"
      variant="borderless"
      style={{ flex: 1 }}
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'Yiminghe', label: 'yiminghe' },
      ]}
    />
    <Select
      mode="multiple"
      defaultValue={['lucy']}
```

```
        placeholder="Borderless"
        variant="borderless"
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
    </Flex>
    <Flex gap={8}>
      <Select
        placeholder="Underlined"
        variant="underlined"
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
      <Select
        mode="multiple"
        defaultValue={['lucy']}
        placeholder="Underlined"
        variant="underlined"
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
    </Flex>
  </Flex>
);

export default App;
```

Filled debug

Debug

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

```tsx
const App: React.FC = () => (
  <Flex gap={12} vertical>
    <Flex gap={8}>
      <Select
        value="lucy"
        disabled
        variant="filled"
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
      <Select
        value="lucy"
        disabled
        mode="multiple"
        variant="filled"
        placeholder="Outlined"
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
    </Flex>
    <Flex gap={8}>
      <Select
        value="lucy"
        status="error"
        variant="filled"
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
      <Select
        value="lucy"
        status="error"
        mode="multiple"
        variant="filled"
        placeholder="Outlined"
```

```
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
    </Flex>
    <Flex gap={8}>
      <Select
        disabled
        value="lucy"
        status="error"
        variant="filled"
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
      <Select
        disabled
        value="lucy"
        status="error"
        mode="multiple"
        variant="filled"
        placeholder="Outlined"
        style={{ flex: 1 }}
        options={[
          { value: 'jack', label: 'Jack' },
          { value: 'lucy', label: 'Lucy' },
          { value: 'Yiminghe', label: 'yiminghe' },
        ]}
      />
    </Flex>
  </Flex>
);

export default App;
```

Custom Tag Render

```
import React from 'react';
import { Select, Tag } from 'antd';
import type { SelectProps } from 'antd';
```

```
type TagRender = SelectProps['tagRender'];

const options: SelectProps['options'] = [
  { value: 'gold' },
  { value: 'lime' },
  { value: 'green' },
  { value: 'cyan' },
];

const tagRender: TagRender = (props) => {
  const { label, value, closable, onClose } = props;
  const onPreventMouseDown = (event: React.MouseEvent<HTMLSpanElement>) =>
{
    event.preventDefault();
    event.stopPropagation();
  };
  return (
    <Tag
      color={value}
      onMouseDown={onPreventMouseDown}
      closable={closable}
      onClose={onClose}
      style={{ marginInlineEnd: 4 }}
    >
      {label}
    </Tag>
  );
};

const App: React.FC = () => (
  <Select
    mode="multiple"
    tagRender={tagRender}
    defaultValue={['gold', 'cyan']}
    style={{ width: '100%' }}
    options={options}
  />
);

export default App;
```

Custom Selected Label Render

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

```tsx
import type { SelectProps } from 'antd';

type LabelRender = SelectProps['labelRender'];

const options = [
  { label: 'gold', value: 'gold' },
  { label: 'lime', value: 'lime' },
  { label: 'green', value: 'green' },
  { label: 'cyan', value: 'cyan' },
];

const labelRender: LabelRender = (props) => {
  const { label, value } = props;

  if (label) {
    return value;
  }
  return <span>No option match</span>;
};

const App: React.FC = () => (
  <Select labelRender={labelRender} defaultValue="1" style={{ width: '100%'
}} options={options} />
);

export default App;
```

Responsive maxTagCount

```tsx
import React, { useState } from 'react';
import type { SelectProps } from 'antd';
import { Select, Space, Tooltip } from 'antd';

interface ItemProps {
  label: string;
  value: string;
}

const options: ItemProps[] = [];

for (let i = 10; i < 36; i++) {
  const value = i.toString(36) + i;
  options.push({
    label: `Long Label: ${value}`,
    value,
  });
```

```
}

const sharedProps: SelectProps = {
  mode: 'multiple',
  style: { width: '100%' },
  options,
  placeholder: 'Select Item...',
  maxTagCount: 'responsive',
};

const App: React.FC = () => {
  const [value, setValue] = useState(['a10', 'c12', 'h17', 'j19', 'k20']);

  const selectProps: SelectProps = {
    value,
    onChange: setValue,
  };

  return (
    <Space direction="vertical" style={{ width: '100%' }}>
      <Select {...sharedProps} {...selectProps} />
      <Select {...sharedProps} disabled />
      <Select
        {...sharedProps}
        {...selectProps}
        maxTagPlaceholder={(omittedValues) => (
          <Tooltip
            styles={{ root: { pointerEvents: 'none' } }}
            title={omittedValues.map(({ label }) => label).join(', ')}
          >
            <span>Hover Me</span>
          </Tooltip>
        )}
      />
    </Space>
  );
};

export default App;
```

Big Data

```
import React from 'react';
import type { SelectProps } from 'antd';
import { Select, Typography } from 'antd';
```

```
const { Title } = Typography;

const options: SelectProps['options'] = [];

for (let i = 0; i < 100000; i++) {
  const value = `${i.toString(36)}${i}`;
  options.push({
    label: value,
    value,
    disabled: i === 10,
  });
}

const handleChange = (value: string[]) => {
  console.log(`selected ${value}`);
};

const App: React.FC = () => (
  <>
    <Title level={4}>{options.length} Items</Title>
    <Select
      mode="multiple"
      style={{ width: '100%' }}
      placeholder="Please select"
      defaultValue={['a10', 'c12']}
      onChange={handleChange}
      options={options}
    />
  </>
);

export default App;
```

Status

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

const App: React.FC = () => (
  <Space direction="vertical" style={{ width: '100%' }}>
    <Select status="error" style={{ width: '100%' }} />
    <Select status="warning" style={{ width: '100%' }} />
  </Space>
);
```

```
export default App;
```

Placement

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

type SelectCommonPlacement = SelectProps['placement'];

const App: React.FC = () => {
  const [placement, SetPlacement] = useState<SelectCommonPlacement>
('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 />
      <Select
        defaultValue="HangZhou"
        style={{ width: 120 }}
        popupMatchSelectWidth={false}
        placement={placement}
        options={[
          {
            value: 'HangZhou',
            label: 'HangZhou #310000',
          },
          {
            value: 'NingBo',
            label: 'NingBo #315000',
          },
          {
            value: 'WenZhou',
            label: 'WenZhou #325000',
```

```
          },
        ]}
      />
    </>
  );
};

export default App;
```

Dynamic Height

Debug

```
import React, { useState } from 'react';
import type { RadioChangeEvent, SelectProps } from 'antd';
import { Button, Radio, Select, Space, Switch } from 'antd';

type SelectCommonPlacement = SelectProps['placement'];

const randomOptions = (count?: number) => {
  const length = count ?? Math.floor(Math.random() * 5) + 1;

  // Random 1 ~ 5 options
  return Array.from({ length }).map((_, index) => ({
    value: index,
    label: `Option ${index}`,
  }));
};

const App: React.FC = () => {
  const [placement, SetPlacement] = useState<SelectCommonPlacement>
('topLeft');
  const [open, setOpen] = useState(false);
  const [options, setOptions] = useState(() => randomOptions(3));

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

  return (
    <div
      style={{
        height: '100%',
        minHeight: 500,
        display: 'flex',
        flexDirection: 'column',
```

```
        justifyContent: 'center',
        alignItems: 'center',
        position: 'relative',
      }}
    >
      <Space
        style={{
          position: 'absolute',
          top: 0,
          insetInlineStart: '50%',
          transform: 'translateX(-50%)',
        }}
      >
        <Radio.Group value={placement} onChange={placementChange}>
          <Radio.Button value="topLeft">TL</Radio.Button>
          <Radio.Button value="topRight">TR</Radio.Button>
          <Radio.Button value="bottomLeft">BL</Radio.Button>
          <Radio.Button value="bottomRight">BR</Radio.Button>
        </Radio.Group>

        <Switch checked={open} onChange={() => setOpen((o) => !o)} />
        <Button onClick={() => setOptions(randomOptions())}>Random</Button>
      </Space>
      <Select
        open={open}
        style={{ width: 120 }}
        placement={placement}
        options={options}
        popupMatchSelectWidth={200}
      />
    </div>
  );
};

export default App;
```

4.0 Debug

Debug

```
import React from 'react';
import { Button, Input, Select, Space } from 'antd';

const style: React.CSSProperties = {
  width: 500,
  position: 'relative',
```

```
  zIndex: 1,
  border: '1px solid red',
  backgroundColor: '#fff',
};

const handleChange = (value: string | string[]) => {
  console.log(`selected ${value}`);
};

const App: React.FC = () => (
  <Space style={style} wrap>
    <Input style={{ width: 100 }} value="222" />
    <Select
      style={{ width: 120 }}
      onChange={handleChange}
      showSearch
      placeholder="233"
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'disabled', disabled: true, label: 'Disabled' },
        { value: 'Yiminghe', label: 'yiminghe' },
        { value: 'long', label: 'I am super super long!' },
      ]}
    />
    <Select
      mode="multiple"
      style={{ width: 120 }}
      defaultValue={['lucy']}
      onChange={handleChange}
      showSearch
      placeholder="233"
      options={[
        { value: 'jack', label: 'Jack' },
        { value: 'lucy', label: 'Lucy' },
        { value: 'disabled', disabled: true, label: 'Disabled' },
        { value: 'Yiminghe', label: 'yiminghe' },
        { value: 'long', label: 'I am super super long!' },
      ]}
    />
    <span className="debug-align">AntDesign</span>
    <Button>222</Button>
  </Space>
);

export default App;
```

_InternalPanelDoNotUseOrYouWillBeFired

Debug

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

const { _InternalPanelDoNotUseOrYouWillBeFired: InternalSelect } = Select;

const App: React.FC = () => {
  const [open, setOpen] = React.useState(true);

  return (
    <Space direction="vertical" style={{ display: 'flex' }}>
      <Switch checked={open} onChange={() => setOpen(!open)} />
      <InternalSelect
        defaultValue="lucy"
        style={{ width: 120 }}
        open={open}
        options={[
          { label: 'Jack', value: 'jack' },
          { label: 'Lucy', value: 'lucy' },
          { label: 'Disabled', value: 'disabled' },
          { label: 'Bamboo', value: 'bamboo' },
        ]}
        virtual={false}
      />
    </Space>
  );
};

export default App;
```

Options label Centered

Debug

```
import React from 'react';
import {
  AutoComplete,
  Cascader,
  Flex,
  Form,
  Input,
  Select,
  Space,
```

```
  TreeSelect,
  Typography,
} from 'antd';

const options = [
  { value: 'long', label: <Typography>long, long, long piece of
text</Typography> },
  { value: 'short', label: <Typography>short</Typography> },
  { value: 'normal', label: <div>normal</div> },
];

const App: React.FC = () => (
  <>
    <Space wrap>
      <Select
        defaultValue="long, long, long piece of text"
        style={{ width: 120 }}
        allowClear
        options={options}
      />

      <Select
        placeholder="Select a option"
        style={{ width: 120, height: 60 }}
        allowClear
        options={options}
      />

      <Select
        defaultValue="normal"
        placeholder="Select a option"
        style={{ width: 120 }}
        allowClear
        options={options}
      />

      <Select
        defaultValue={['normal']}
        mode="multiple"
        placeholder="Select a option"
        style={{ width: 120 }}
        allowClear
        options={options}
      />

      <Select
```

```
  mode="multiple"
  placeholder="Select a option"
  style={{ width: 120, height: 60 }}
  allowClear
  options={options}
/>

<Cascader
  placeholder="Select a option"
  style={{ width: 120, height: 60 }}
  allowClear
  options={options}
/>

<TreeSelect
  showSearch
  style={{ width: 120, height: 60 }}
  placeholder="Please select"
  allowClear
  popupMatchSelectWidth={false}
  treeDefaultExpandAll
  treeData={[
    {
      value: 'parent 1',
      title: 'parent 1',
      children: options,
    },
  ]}
/>
<Select
  prefix="Hello World"
  mode="multiple"
  allowClear
  placeholder="Select"
  style={{ minWidth: 200, height: 200 }}
  defaultValue={['long']}
  options={options}
/>
<Select
  mode="multiple"
  style={{ width: 200 }}
  placeholder="请选择"
  maxTagCount="responsive"
  prefix="城市"
  options={options}
/>
```

```
<Select
  style={{ width: 200 }}
  placeholder="请选择"
  prefix="城市"
  options={options}
  showSearch
  allowClear
  status="error"
/>
<Select
  style={{ width: 100 }}
  prefix="Hi"
  options={options}
  showSearch
  allowClear
  status="warning"
  variant="filled"
  defaultValue="Bamboo"
/>
<Select
  style={{ width: 100 }}
  prefix="Hi"
  options={options}
  showSearch
  allowClear
  status="error"
  variant="borderless"
  defaultValue="Bamboo"
/>
<Form style={{ width: 200 }} layout="vertical">
  <Form.Item
    label="Label"
    name="bamboo"
    initialValue="Bamboo"
    style={{
      boxShadow: '0 0 0 1px red',
    }}
  >
    <Select options={options} allowClear placeholder="bamboo" />
  </Form.Item>
  <Form.Item
    label="Label"
    name="bamboo"
    initialValue="Bamboo"
    style={{
      boxShadow: '0 0 0 1px red',
```

```
          }}
        >
          <AutoComplete options={options} allowClear placeholder="bamboo"
/>
        </Form.Item>
      </Form>
    </Space>

    {/* Test for align */}
    <Flex vertical style={{ width: 200 }}>
      {/* Single */}
      <Input prefix="Hi" placeholder="Input" allowClear />
      <Select prefix="Hi" placeholder="Single" options={options} allowClear
showSearch />
      <Select
        prefix="Hi"
        placeholder="Single"
        options={options}
        allowClear
        showSearch
        defaultValue="Bamboo"
      />
      {/* Multiple */}
      <Select placeholder="Multiple" options={options} allowClear
mode="multiple" />
      <Select prefix="Hi" placeholder="Multiple" options={options}
allowClear mode="multiple" />
      <Select
        prefix="Hi"
        placeholder="Multiple"
        options={options}
        allowClear
        mode="multiple"
        defaultValue={['Bamboo']}
      />
      <Select
        placeholder="Multiple"
        options={options}
        allowClear
        mode="multiple"
        defaultValue={['Bamboo']}
      />
    </Flex>
  </>
);
```

```
export default App;
```

Flip + Shift

Debug

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

const App: React.FC = () => (
  <Select
    style={{ width: 120, marginTop: '50vh' }}
    open
    options={Array.from({ length: 100 }).map((_, index) => ({
      value: index,
    }))}
  />
);

export default App;
```

Component Token

Debug

```
import React from 'react';
import { ConfigProvider, Select, Space } from 'antd';
import type { SelectProps } from 'antd';

const options: SelectProps['options'] = [];

for (let i = 10; i < 36; i++) {
  options.push({
    label: i.toString(36) + i,
    value: i.toString(36) + i,
  });
}

const App: React.FC = () => (
  <Space direction="vertical">
    <ConfigProvider
      theme={{
        components: {
          Select: {
            multipleItemBorderColor: 'rgba(0,0,0,0.06)',
```

```
          multipleItemBorderColorDisabled: 'rgba(0,0,0,0.06)',
          optionSelectedColor: '#1677ff',
          hoverBorderColor: 'red',
          activeBorderColor: 'green',
          activeOutlineColor: 'pink',
        },
      },
    }}
  >
    <Space style={{ width: '100%' }} direction="vertical">
      <Select
        mode="multiple"
        allowClear
        style={{ width: '100%' }}
        placeholder="Please select"
        defaultValue={['a10', 'c12']}
        options={options}
      />
      <Select
        mode="multiple"
        disabled
        style={{ width: '100%' }}
        placeholder="Please select"
        defaultValue={['a10', 'c12']}
        options={options}
      />
    </Space>
  </ConfigProvider>
  <ConfigProvider
    theme={{
      token: {
        controlHeightSM: 28,
      },
    }}
  >
    <Space style={{ width: '100%' }} direction="vertical">
      <Select
        mode="multiple"
        allowClear
        size="small"
        style={{ width: '100%' }}
        placeholder="Please select"
        defaultValue={['a10', 'c12']}
        options={options}
      />
      <Select
```

```
          mode="multiple"
          allowClear
          style={{ width: '100%' }}
          placeholder="Please select"
          defaultValue={['a10', 'c12']}
          options={options}
        />
      </Space>
    </ConfigProvider>
    <ConfigProvider
      theme={{
        components: {
          Select: {
            paddingXXS: 0,
            controlHeight: 28,
          },
        },
      }}
    >
      <Space style={{ width: '100%' }} direction="vertical">
        <Select style={{ width: '100%' }} defaultValue="a10" options=
{options} />
        <Select
          mode="multiple"
          style={{ width: '100%' }}
          defaultValue={['a10', 'c12']}
          options={options}
        />
      </Space>
    </ConfigProvider>
  </Space>
);

export default App;
```

Max Count

v5.13.0

```
import React from 'react';
import { DownOutlined } from '@ant-design/icons';
import { Select } from 'antd';

const MAX_COUNT = 3;

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

```
  const [value, setValue] = React.useState<string[]>(['Ava Swift']);

  const suffix = (
    <>
      <span>
        {value.length} / {MAX_COUNT}
      </span>
      <DownOutlined />
    </>
  );

  return (
    <Select
      mode="multiple"
      maxCount={MAX_COUNT}
      value={value}
      style={{ width: '100%' }}
      onChange={setValue}
      suffixIcon={suffix}
      placeholder="Please select"
      options={[
        { value: 'Ava Swift', label: 'Ava Swift' },
        { value: 'Cole Reed', label: 'Cole Reed' },
        { value: 'Mia Blake', label: 'Mia Blake' },
        { value: 'Jake Stone', label: 'Jake Stone' },
        { value: 'Lily Lane', label: 'Lily Lane' },
        { value: 'Ryan Chase', label: 'Ryan Chase' },
        { value: 'Zoe Fox', label: 'Zoe Fox' },
        { value: 'Alex Grey', label: 'Alex Grey' },
        { value: 'Elle Blair', label: 'Elle Blair' },
      ]}
    />
  );
};

export default App;
```

API

Common props ref：Common props

Select props

Property	Description	Type	
allowClear	Customize clear icon	boolean | { clearIcon?: ReactNode }	fal

autoClearSearchValue	Whether the current search will be cleared on selecting an item. Only applies when `mode` is set to `multiple` or `tags`	boolean	tru
autoFocus	Get focus by default	boolean	fal
defaultActiveFirstOption	Whether active first option by default	boolean	tru
defaultOpen	Initial open state of dropdown	boolean	-
defaultValue	Initial selected option	string | string[] | number | number[] | LabeledValue | LabeledValue[]	-
disabled	Whether disabled select	boolean	fal
popupClassName	The className of dropdown menu	string	-
popupMatchSelectWidth	Determine whether the popup menu and the select input are the same width. Default set `min-width` same as input. Will ignore when value less than select width. `false` will disable virtual scroll	boolean | number	tru
dropdownRender	Customize dropdown content	(originNode: ReactNode) => ReactNode	-
dropdownStyle	The style of dropdown menu	CSSProperties	-

fieldNames	Customize node label, value, options, groupLabel field name	object	{ la val op op gro la
filterOption	If true, filter options by input, if function, filter options against it. 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	boolean | function(inputValue, option)	tru
filterSort	Sort function for search options sorting, see Array.sort's compareFunction	(optionA: Option, optionB: Option, info: { searchValue: string }) => number	-
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)	() = do
labelInValue	Whether to embed label in value, turn the format of value from `string` to { value: string, label: ReactNode }	boolean	fal

listHeight	Config popup height	number	25
loading	Indicate loading state	boolean	fal
maxCount	The max number of items can be selected, only applies when `mode` is `multiple` or `tags`	number	-
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	-
menuItemSelectedIcon	The custom menuItemSelected icon with multiple options	ReactNode	-
mode	Set mode of Select	`multiple` | `tags`	-
notFoundContent	Specify content to show when no result matches	ReactNode	No
open	Controlled open state of dropdown	boolean	-
optionFilterProp	Which prop value of option will be used for filter if filterOption is true. If `options` is set, it should be set to `label`	string	va
optionLabelProp	Which prop value of option will	string	ch

	render as content of select. Example		
options	Select options. Will get better perf than jsx definition	{ label, value }[]	-
optionRender	Customize the rendering dropdown options	(option: FlattenOptionData<BaseOptionType> , info: { index: number }) => React.ReactNode	-
placeholder	Placeholder of select	ReactNode	-
placement	The position where the selection box pops up	`bottomLeft` `bottomRight` `topLeft` `topRight`	bo
prefix	The custom prefix	ReactNode	-
removeIcon	The custom remove icon	ReactNode	-
searchValue	The current input "search" text	string	-
showSearch	Whether select is searchable	boolean	sir mu
size	Size of Select input	`large` | `middle` | `small`	mi
status	Set validation status	'error' | 'warning'	-
suffixIcon	The custom suffix icon. Customize icon will not response click open to avoid icon designed to do other interactive. You can use `pointer-events: none` style to bypass	ReactNode	<D />

tagRender	Customize tag render, only applies when `mode` is set to `multiple` or `tags`	(props) => ReactNode	-
labelRender	Customize selected label render (LabelInValueType definition see LabelInValueType)	(props: LabelInValueType) => ReactNode	-
tokenSeparators	Separator used to tokenize, only applies when `mode="tags"`	string[]	-
value	Current selected option (considered as a immutable array)	string | string[] | number | number[] | LabeledValue | LabeledValue[]	-
variant	Variants of selector	`outlined` | `borderless` | `filled` | `underlined`	ou
virtual	Disable virtual scroll when set to false	boolean	tru
onBlur	Called when blur	function	-
onChange	Called when select an option or input value change	function(value, option:Option | Array<Option>)	-
onClear	Called when clear	function	-
onDeselect	Called when an option is deselected, param is the selected option's value. Only called for `multiple` or `tags`, effective in	function(value: string | number | LabeledValue)	-

	multiple or tags mode only		
onDropdownVisibleChange	Called when dropdown open	(open: boolean) => void	-
onFocus	Called when focus	(event: FocusEvent) => void	-
onInputKeyDown	Called when key pressed	(event: KeyboardEvent) => void	-
onPopupScroll	Called when dropdown scrolls	(event: UIEvent) => void	-
onSearch	Callback function that is fired when input changed	function(value: string)	-
onSelect	Called when an option is selected, the params are option's value (or key) and option instance	function(value: string | number | LabeledValue, option: Option)	-

> *Note, if you find that the drop-down menu scrolls with the page, or you need to trigger Select in other popup layers, please try to use* `getPopupContainer={triggerNode => triggerNode.parentElement}` *to fix the drop-down popup rendering node in the parent element of the trigger .*

Select Methods

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

Option props

Property	Description	Type	Default	Version
className	The additional class to option	string	-	
disabled	Disable this option	boolean	false	
title	`title` attribute of Select Option	string	-	
value	Default to filter with this property	string | number	-	

OptGroup props

Property	Description	Type	Default	Version
key	Group key	string	-	
label	Group label	React.ReactNode	-	
className	The additional class to option	string	-	
title	`title` attribute of Select Option	string	-	

Design Token

FAQ

Why sometime search will show 2 same option when in `tags` mode?

It's caused by option with different `label` and `value`. You can use `optionFilterProp="label"` to change filter logic instead.

When I click elements in dropdownRender, the select dropdown will not be closed?

You can control it by `open` prop: codesandbox.

I don't want dropdown close when click inside `dropdownRender`?

Select will close when it lose focus. You can prevent event to handle this:

```
<Select
  dropdownRender={() => (
    <div
      onMouseDown={(e) => {
        e.preventDefault();
        e.stopPropagation();
      }}
    >
      Some Content
    </div>
  )}
/>
```

Why sometime customize Option cause scroll break?

Virtual scroll internal set item height as `24px`. You need to adjust `listItemHeight` when your option height is less and `listHeight` config list container height:

```
<Select listItemHeight={10} listHeight={250} />
```

Note: `listItemHeight` and `listHeight` are internal props. Please only modify when necessary.

Why a11y test report missing `aria-` props?

Select only create a11y auxiliary node when operating on. Please open Select and retry. For `aria-label` & `aria-labelledby` miss warning, please add related prop to Select with your own requirement.

Default virtual scrolling will create a mock element to simulate an accessible binding. If a screen reader needs to fully access the entire list, you can set `virtual={false}` to disable virtual scrolling and the accessibility option will be bound to the actual element.