AutoComplete ✎

Autocomplete function of input field.

When To Use

- When you need an input box instead of a selector.
- When you need input suggestions or helping text.

The differences with Select are:

- AutoComplete is an input box with text hints, and users can type freely. The keyword is aiding **input**.
- Select is selecting among given choices. The keyword is **select**.

Examples

input here

control mode

Basic Usage, set data source of autocomplete with `options` property.

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

const mockVal = (str: string, repeat = 1) => (
  value: str.repeat(repeat),
});

const App: React.FC = () => {
  const [value, setValue] = useState('');
  const [options, setOptions] = useState<{ val
  const [anotherOptions, setAnotherOptions] =

  const getPanelValue = (searchText: string) =
    !searchText ? [] : [mockVal(searchText), m

  const onSelect = (data: string) => {
    console.log('onSelect', data);
  };

  const onChange = (data: string) => {
    setValue(data);
  };

  return (
    <>
      <AutoComplete
        options={options}
        style={{ width: 200 }}
        onSelect={onSelect}
        onSearch={(text) => setOptions(getPane
        placeholder="input here"
      />
      <br />
      <br />
      <AutoComplete
        value={value}
        options={anotherOptions}
        style={{ width: 200 }}
        onSelect={onSelect}
        onSearch={(text) => setAnotherOptions(
        onChange={onChange}
        placeholder="control mode"
      />
    </>
  );
};

export default App;
```

input here

You could set custom `Option` label

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

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

  const handleSearch = (value: string) => {
    let res: { value: string; label: string }[
    if (!value || value.indexOf('@') >= 0) {
      res = [];
    } else {
      res = ['gmail.com', '163.com', 'qq.com']
        value,
        label: `${value}@${domain}`,
      }));
    }
    setOptions(res);
  };

  return (
    <AutoComplete
      style={{ width: 200 }}
      onSearch={handleSearch}
      placeholder="input here"
      options={options}
    />
  );
};

export default App;
```

input here

Customize Input Component ✎

Customize Input Component

```tsx
import React, { useState } from 'react';
import { AutoComplete, Input } from 'antd';

const { TextArea } = Input;

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

  const handleSearch = (value: string) => {
    setOptions(
      !value ? [] : [{ value }, { value: value
    );
  };

  const handleKeyPress = (ev: React.KeyboardEv
    console.log('handleKeyPress', ev);
  };

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

  return (
    <AutoComplete
      options={options}
      style={{ width: 200 }}
      onSelect={onSelect}
      onSearch={handleSearch}
    >
      <TextArea
        placeholder="input here"
        className="custom"
        style={{ height: 50 }}
        onKeyPress={handleKeyPress}
      />
    </AutoComplete>
  );
};

export default App;
```

try to type `b`

Non-case-sensitive AutoComplete ✎

A non-case-sensitive AutoComplete

```tsx
import React from 'react';
import { AutoComplete } from 'antd';

const options = [
  { value: 'Burns Bay Road' },
  { value: 'Downing Street' },
  { value: 'Wall Street' },
];

const App: React.FC = () => (
  <AutoComplete
    style={{ width: 200 }}
    options={options}
    placeholder="try to type `b`"
    filterOption={(inputValue, option) =>
      option!.value.toUpperCase().indexOf(inpu
    }
  />
);

export default App;
```

| input here | 🔍 |

| input here |

Lookup-Patterns - Certain Category ✎

Demonstration of [Lookup Patterns: Certain Category](). Basic Usage, set options of autocomplete with `options` property.

```
import React from 'react';
import { UserOutlined } from '@ant-design/icor
import { AutoComplete, Input } from 'antd';

const renderTitle = (title: string) => (
  <span>
    {title}
    <a
      style={{ float: 'right' }}
      href="https://www.google.com/search?q=ar
      target="_blank"
      rel="noopener noreferrer"
    >
      more
    </a>
  </span>
);

const renderItem = (title: string, count: numt
  value: title,
  label: (
    <div
      style={{
        display: 'flex',
        justifyContent: 'space-between',
      }}
    >
      {title}
      <span>
        <UserOutlined /> {count}
      </span>
    </div>
  ),
});

const options = [
  {
    label: renderTitle('Libraries'),
    options: [renderItem('AntDesign', 10000),
  },
  {
    label: renderTitle('Solutions'),
    options: [renderItem('AntDesign UI FAQ', 6
  },
  {
    label: renderTitle('Articles'),
    options: [renderItem('AntDesign design lar
  },
];

const App: React.FC = () => (
  <AutoComplete
    popupClassName="certain-category-search-di
    dropdownMatchSelectWidth={500}
    style={{ width: 250 }}
    options={options}
  >
    <Input.Search size="large" placeholder="ir
  </AutoComplete>
);
```

Lookup-Patterns - Uncertain Category ✎

Demonstration of [Lookup Patterns: Uncertain Category]().

```
import React, { useState } from 'react';
import { AutoComplete, Input } from 'antd';
import type { SelectProps } from 'antd/es/sele

const getRandomInt = (max: number, min = 0) =>

const searchResult = (query: string) =>
  new Array(getRandomInt(5))
    .join('.')
    .split('.')
    .map((_, idx) => {
      const category = `${query}${idx}`;
      return {
        value: category,
        label: (
          <div
            style={{
              display: 'flex',
              justifyContent: 'space-between',
            }}
          >
            <span>
              Found {query} on{' '}
              <a
                href={`https://s.taobao.com/se
                target="_blank"
                rel="noopener noreferrer"
              >
                {category}
              </a>
            </span>
            <span>{getRandomInt(200, 100)} res
          </div>
        ),
      };
    });

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

  const handleSearch = (value: string) => {
    setOptions(value ? searchResult(value) : [
  };

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

  return (
    <AutoComplete
      dropdownMatchSelectWidth={252}
      style={{ width: 300 }}
      options={options}
      onSelect={onSelect}
      onSearch={handleSearch}
    >
      <Input.Search size="large" placeholder='
    </AutoComplete>
  );
};

export default App;
```

Status

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

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

const mockVal = (str: string, repeat = 1) => (
  value: str.repeat(repeat),
});

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

  const getPanelValue = (searchText: string) =
    !searchText ? [] : [mockVal(searchText), m

  return (
    <Space direction="vertical" style={{ width
      <AutoComplete
        options={options}
        onSearch={(text) => setOptions(getPane
        status="error"
        style={{ width: 200 }}
      />
      <AutoComplete
        options={anotherOptions}
        onSearch={(text) => setAnotherOptions(
        status="warning"
        style={{ width: 200 }}
      />
    </Space>
  );
};

export default App;
```

API

Property	Description	Type
allowClear	Show clear button	boolean
autoFocus	If get focus when component mounted	boolean
backfill	If backfill selected item the input when using keyboard	boolean
children (for customize input element)	Customize input element	HTMLInputElement | HTML | React.ReactElement<Inp
children (for dataSource)	Data source to auto complete	React.ReactElement<Optic Array<React.ReactElement

Property	Description	Type
defaultActiveFirstOption	Whether active first option by default	boolean
defaultOpen	Initial open state of dropdown	boolean
defaultValue	Initial selected option	string
disabled	Whether disabled select	boolean
popupClassName	The className of dropdown menu	string
dropdownMatchSelectWidth	Determine whether the dropdown 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
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(inpu
notFoundContent	Specify content to show when no result matches	string
open	Controlled open state of dropdown	boolean
options	Select options. Will get better perf than jsx definition	{ label, value }[]
placeholder	The placeholder of input	string
status	Set validation status	'error' | 'warning'
value	Selected option	string

Property	Description	Type
onBlur	Called when leaving the component	function()
onChange	Called when selecting an option or changing an input value	function(value)
onDropdownVisibleChange	Call when dropdown open	function(open)
onFocus	Called when entering the component	function()
onSearch	Called when searching items	function(value)
onSelect	Called when a option is selected. param is option's value and option instance	function(value, option)
onClear	Called when clear	function

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)
colorBgElevated	Container background color of the popup layer, in dark mode the color value of this token will be a little brighter than `colorBgContainer`. E.g: modal, pop-up, menu, etc.	string	☐ #ffffff
colorBorder	Default border color, used to separate different elements, such as: form separator, card separator, etc.	string	☐ #d9d9d9
colorErrorHover	The hover state of the error color.	string	☐ #ff7875

Token Name	Description	Type	Default Value
colorErrorOutline	Control the outline color of input component in error state.	string	☐ rgba(255, 38, 5, 0.06)
colorFillSecondary	The second level of fill color can outline the shape of the element more clearly, such as Rate, Skeleton, etc. It can also be used as the Hover state of the third level of fill color, such as Table, etc.	string	☐ rgba(0, 0, 0, 0.06)
colorIcon	Weak action. Such as `allowClear` or Alert close button	string	☐ rgba(0, 0, 0, 0.45)
colorIconHover	Weak action hover color. Such as `allowClear` or Alert close button	string	☐ rgba(0, 0, 0, 0.88)
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
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 with transparency.	string	☐ rgba(5, 5, 5, 0.06)
colorText	Default text color which comply with W3C standards, and this color is also the darkest neutral color.	string	☐ rgba(0, 0, 0, 0.88)
colorTextDescription	Control the font color of text description.	string	☐ rgba(0, 0, 0, 0.45)
colorTextDisabled	Control the color of text in disabled state.	string	☐ rgba(0, 0, 0, 0.25)
colorTextPlaceholder	Control the color of placeholder text.	string	☐ rgba(0, 0, 0, 0.25)
colorTextQuaternary	The fourth level of text color is the lightest text color, such as form input prompt text, disabled color text, etc.	string	☐ rgba(0, 0, 0, 0.25)
colorTextTertiary	The third level of text color is generally used for descriptive text, such as form supplementary explanation text, list descriptive text, etc.	string	☐ rgba(0, 0, 0, 0.45)
colorWarningHover	The hover state of the warning color.	string	☐ #ffd666
colorWarningOutline	Control the outline color of input component in warning state.	string	☐ rgba(255, 215, 5, 0.1)
borderRadius	Border radius of base components	number	6

Token Name	Description	Type	Default Value
borderRadiusLG	LG size border radius, used in some large border radius components, such as Card, Modal and other components.	number	8
borderRadiusSM	SM size border radius, used in small size components, such as Button, Input, Select and other input components in small size	number	4
borderRadiusXS	XS size border radius, used in some small border radius components, such as Segmented, Arrow and other components with small border radius.	number	2
boxShadowSecondary	Control the secondary box shadow style of an element.	string	0 6px 16px 0 rgba(0, 0, 0, 0.08), 0 3px 6px -4px rgba(0, 0, 0, 0.12), 0 9px 28px 8px rgba(0, 0, 0, 0.05)
controlHeight	The height of the basic controls such as buttons and input boxes in Ant Design	number	32
controlHeightLG	LG component height	number	40
controlHeightSM	SM component height	number	24
controlHeightXS	XS component height	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)
controlOutline	Control the outline color of input component.	string	☐ rgba(5, 145, 255, 0.1)
controlOutlineWidth	Control the outline width of input component.	number	2
controlPaddingHorizontal	Control the horizontal padding of an element.	number	12
controlPaddingHorizontalSM	Control the horizontal padding of an element with a small-medium size.	number	8
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

Token Name	Description	Type	Default Value
fontSizeIcon	Control the font size of operation icon in Select, Cascader, etc. Normally same as fontSizeSM.	`number`	12
fontSizeLG	Large font size	`number`	16
fontSizeSM	Small font size	`number`	12
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
motionDurationMid	Motion speed, medium speed. Used for medium element animation interaction.	`string`	0.2s
motionDurationSlow	Motion speed, slow speed. Used for large element animation interaction.	`string`	0.3s
motionEaseInOut	Preset motion curve.	`string`	cubic-bezier(0.645, 0.045, 0.355, 1)
motionEaseInOutCirc	Preset motion curve.	`string`	cubic-bezier(0.78, 0.14, 0.15, 0.86)
motionEaseInQuint	Preset motion curve.	`string`	cubic-bezier(0.755, 0.05, 0.855, 0.06)
motionEaseOutCirc	Preset motion curve.	`string`	cubic-bezier(0.08, 0.82, 0.17, 1)
motionEaseOutQuint	Preset motion curve.	`string`	cubic-bezier(0.23, 1, 0.32, 1)
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
zIndexPopupBase	Base zIndex of component like FloatButton, Affix which can be cover by large popup	`number`	1000

FAQ

Why doesn't the text composition system work well with onSearch in controlled mode?

Please use `onChange` to manage control state. `onSearch` is used for searching input which is not the same as `onChange`. Besides, clicking on the option will not trigger the `onSearch` event.

Related issue: [#18230](#) [#17916](#)