Mentions

Mention component.

When To Use

When you need to mention someone or something.

Usage upgrade after 5.1.0

After version 5.1.0, we provide a simpler usage <Mentions 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.

```jsx
// works when >=5.1.0, recommended ✅
const options = [{ value: 'sample', label: 'sample' }];
return <Mentions options={options} />;

// works when <5.1.0, deprecated when >=5.1.0 🙅🏻‍♀️
return (
  <Mentions onChange={onChange}>
    <Mentions.Option value="sample">Sample</Mentions.Option>
  </Mentions>
);
```

Examples

@afc163

Basic

Basic usage.

```
import React from 'react';
import { Mentions } from 'antd';
import type { MentionsOptionProps } from 'antc

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

const onSelect = (option: MentionsOptionProps)
  console.log('select', option);
};

const App: React.FC = () => (
  <Mentions
    style={{ width: '100%' }}
    onChange={onChange}
    onSelect={onSelect}
    defaultValue="@afc163"
    options={[
      {
        value: 'afc163',
        label: 'afc163',
      },
      {
        value: 'zombieJ',
        label: 'zombieJ',
      },
      {
        value: 'yesmeck',
        label: 'yesmeck',
      },
    ]}
  />
);

export default App;
```

Asynchronous loading

async.

```
import React, { useCallback, useRef, useState
import { Mentions } from 'antd';
import debounce from 'lodash/debounce';

const App: React.FC = () => {
  const [loading, setLoading] = useState(false
  const [users, setUsers] = useState<{ login:
  const ref = useRef<string>();

  const loadGithubUsers = (key: string) => {
    if (!key) {
      setUsers([]);
      return;
    }

    fetch(`https://api.github.com/search/users
      .then((res) => res.json())
      .then(({ items = [] }) => {
        if (ref.current !== key) return;

        setLoading(false);
        setUsers(items.slice(0, 10));
      });
  };

  const debounceLoadGithubUsers = useCallback(

  const onSearch = (search: string) => {
    console.log('Search:', search);
    ref.current = search;
    setLoading(!!search);
    setUsers([]);

    debounceLoadGithubUsers(search);
  };

  return (
    <Mentions
      style={{ width: '100%' }}
      loading={loading}
      onSearch={onSearch}
      options={users.map(({ login, avatar_url:
        key: login,
        value: login,
        className: 'antd-demo-dynamic-option',
        label: (
          <>
            <img src={avatar} alt={login} />
            <span>{login}</span>
          </>
        ),
      }))}
    />
  );
};

export default App;
```

Top coders:

* Bio: You can use @ to ref user here

Submit Reset

With Form

Controlled mode, for example, to work with `Form`.

```
import { Button, Form, Mentions, Space } from
import React from 'react';

const { getMentions } = Mentions;

const App: React.FC = () => {
  const [form] = Form.useForm();

  const onReset = () => {
    form.resetFields();
  };

  const onFinish = async () => {
    try {
      const values = await form.validateFields
      console.log('Submit:', values);
    } catch (errInfo) {
      console.log('Error:', errInfo);
    }
  };

  const checkMention = async (_: any, value: s
    const mentions = getMentions(value);

    if (mentions.length < 2) {
      throw new Error('More than one must be s
    }
  };

  return (
    <Form form={form} layout="horizontal" onFi
      <Form.Item
        name="coders"
        label="Top coders"
        labelCol={{ span: 6 }}
        wrapperCol={{ span: 16 }}
        rules={[{ validator: checkMention }]}
      >
        <Mentions
          rows={1}
          options={[
            {
              value: 'afc163',
              label: 'afc163',
            },
            {
              value: 'zombieJ',
              label: 'zombieJ',
            },
            {
              value: 'yesmeck',
              label: 'yesmeck',
            },
          ]}
        />
```

input @ to mention people, # to mention tag

Customize Trigger Token

Customize Trigger Token by `prefix` props.

Default to `@`, `Array<string>` also supported.

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

const MOCK_DATA = {
  '@': ['afc163', 'zombiej', 'yesmeck'],
  '#': ['1.0', '2.0', '3.0'],
};

type PrefixType = keyof typeof MOCK_DATA;

const App: React.FC = () => {
  const [prefix, setPrefix] = useState<PrefixT

  const onSearch = (_: string, newPrefix: Pre
    setPrefix(newPrefix);
  };

  return (
    <Mentions
      style={{ width: '100%' }}
      placeholder="input @ to mention people,
      prefix={['@', '#']}
      onSearch={onSearch}
      options={(MOCK_DATA[prefix] || []).map((
        key: value,
        value,
        label: value,
      }))}
    />
  );
};

export default App;
```

this is disabled Mentions

this is readOnly Mentions

disabled or readOnly

Configure `disabled` and `readOnly`.

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

const options = ['afc163', 'zombiej', 'yesmeck
  value,
  key: value,
  label: value,
}));

const App: React.FC = () => (
  <>
    <div style={{ marginBottom: 10 }}>
      <Mentions
        style={{ width: '100%' }}
        placeholder="this is disabled Mentions
        disabled
        options={options}
      />
    </div>
    <Mentions
      style={{ width: '100%' }}
      placeholder="this is readOnly Mentions"
      readOnly
      options={options}
    />
  </>
);

export default App;
```

Placement

Change the suggestions placement.

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

const App: React.FC = () => (
  <Mentions
    style={{ width: '100%' }}
    placement="top"
    options={[
      {
        value: 'afc163',
        label: 'afc163',
      },
      {
        value: 'zombieJ',
        label: 'zombieJ',
      },
      {
        value: 'yesmeck',
        label: 'yesmeck',
      },
    ]}
  />
);

export default App;
```

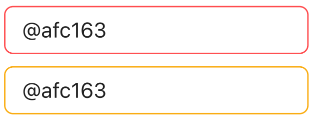
autoSize

Height autoSize.

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

const App: React.FC = () => (
  <Mentions
    autoSize
    style={{ width: '100%' }}
    options={[
      {
        value: 'afc163',
        label: 'afc163',
      },
      {
        value: 'zombieJ',
        label: 'zombieJ',
      },
      {
        value: 'yesmeck',
        label: 'yesmeck',
      },
    ]}
  />
);

export default App;
```



Status

Add status to Mentions with `status`, which could be `error` or `warning`。

```
import React from 'react';
import { Mentions, Space } from 'antd';
import type { MentionsOptionProps } from 'ant

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

const onSelect = (option: MentionsOptionProps)
  console.log('select', option);
};

const App: React.FC = () => {
  const options = [
    {
      value: 'afc163',
      label: 'afc163',
    },
    {
      value: 'zombieJ',
      label: 'zombieJ',
    },
    {
      value: 'yesmeck',
      label: 'yesmeck',
    },
  ];

  return (
    <Space direction="vertical">
      <Mentions
        onChange={onChange}
        onSelect={onSelect}
        defaultValue="@afc163"
        status="error"
        options={options}
      />
      <Mentions
        onChange={onChange}
        onSelect={onSelect}
        defaultValue="@afc163"
        status="warning"
        options={options}
      />
    </Space>
  );
};

export default App;
```

API

Mention

Property	Description	Type	Default	Version
autoFocus	Auto get focus when component mounted	boolean	false	
autoSize	Textarea height autosize feature, can be set to true | false or an object { minRows: 2, maxRows: 6 }	boolean | object	false	
defaultValue	Default value	string	-	
filterOption	Customize filter option logic	false | (input: string, option: OptionProps) => boolean	-	
getPopupContainer	Set the mount HTML node for suggestions	() => HTMLElement	-	
notFoundContent	Set mentions content when not match	ReactNode	Not Found	
placement	Set popup placement	top | bottom	bottom	
prefix	Set trigger prefix keyword	string | string[]	@	
split	Set split string before and after selected mention	string		
status	Set validation status	'error' | 'warning' | 'success' | 'validating'	-	4.19.0
validateSearch	Customize trigger search logic	(text: string, props: MentionsProps) => void	-	
value	Set value of mentions	string	-	
onBlur	Trigger when mentions lose focus	() => void	-	
onChange	Trigger when value changed	(text: string) => void	-	
onFocus	Trigger when mentions get focus	() => void	-	
onResize	The callback function that is triggered when textarea resize	function({ width, height })	-	
onSearch	Trigger when prefix hit	(text: string, prefix:	-	

Property	Description	Type	Default	Version
		string) => void		
onSelect	Trigger when user select the option	(option: OptionProps, prefix: string) => void	-	
options	Option Configuration	[Options]()	[]	5.1.0

Mention methods

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

Option

Property	Description	Type	Default
label	Title of the option	React.ReactNode	-
key	The key value of the option	string	-
disabled	Optional	boolean	-
className	className	string	-
style	The style of the option	React.CSSProperties	-

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

Token Name	Description	Type	Default Value
colorError	Used to represent the visual elements of the operation failure, such as the error Button, error Result component, etc.	`string`	□ `#ff4d4f`
colorErrorBorderHover	The hover state border color of the error state.	`string`	□ `#ffa39e`
colorErrorOutline	Control the outline color of input component in error state.	`string`	□ `rgba(255, 38, 5, 0.06)`
colorPrimaryHover	Hover state under the main color gradient.	`string`	□ `#4096ff`
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)`
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)`
colorWarning	Used to represent the warning map token, such as Notification, Alert, etc. Alert or Control component(like Input) will use these map tokens.	`string`	□ `#faad14`
colorWarningBorderHover	The hover state border color of the warning state.	`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
borderRadiusLG	LG size border radius, used in some large border radius components, such as Card, Modal and other components.	`number`	8
borderRadiusSM	SM size border radius, used in small size components, such as Button, Input, Select and other input components in small size	`number`	4
boxShadowSecondary	Control the secondary box shadow style of an element.	`string`	0 6px 16px 0 rgba(0, 0, 0, 0.08), 0 3px 6px -4px rgba(0, 0, 0, 0.12), 0 9px 28px 8px rgba(0, 0, 0, 0.05)
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
controlItemBgHover	Control the background color of control	`string`	□ `rgba(0, 0, 0, 0.04)`

Token Name	Description	Type	Default Value
	component item when hovering.		
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
fontSizeLG	Large font size	`number`	16
fontWeightStrong	Control the font weight of heading components (such as h1, h2, h3) or selected item.	`number`	600
lineHeight	Line height of text.	`number`	1.5714285714285714
lineHeightLG	Line height of large text.	`number`	1.5
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
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