何时使用 {#when-to-use}

用于在输入中提及某人或某事，常用于发布、聊天或评论功能。

用法升级 5.1.0+

:::warning{title="升级提示"} 在 5.1.0 版本后，我们提供了 `<Mentions options={[...]} />` 的简写方式，有更好的性能和更方便的数据组织方式，开发者不再需要自行拼接 JSX。 同时我们废弃了原先的写法，你还是可以在 5.x 继续使用，但会在控制台看到警告，并会在 6.0 后移除。 :::

```
// >=5.1.0 可用，推荐的写法 ✅
const options = [{ value: 'sample', label: 'sample' }];
return <Mentions options={options} />;

// <5.1.0 可用，>=5.1.0 时不推荐 🤦‍♀️
return (
  <Mentions onChange={onChange}>
    <Mentions.Option value="sample">Sample</Mentions.Option>
  </Mentions>
);
```

代码演示

基本使用

```
import React from 'react';
import { Mentions } from 'antd';
import type { GetProp, MentionProps } from 'antd';

type MentionsOptionProps = GetProp<MentionProps, 'options'>[number];

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;
```

形态变体

v5.13.0

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

const App: React.FC = () => (
  <Flex vertical gap={12}>
    <Mentions placeholder="Outlined" />
    <Mentions placeholder="Filled" variant="filled" />
    <Mentions placeholder="Borderless" variant="borderless" />
    <Mentions placeholder="Underlined" variant="underlined" />
  </Flex>
);

export default App;
```

异步加载

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

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

```
  const ref = useRef<string>(null);

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

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

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

  const debounceLoadGithubUsers = useCallback(debounce(loadGithubUsers,
800), []);

  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: avatar }) => ({
        key: login,
        value: login,
        className: 'antd-demo-dynamic-option',
        label: (
          <>
            <img src={avatar} alt={login} />
            <span>{login}</span>
          </>
        ),
      }))}
```

```
    />
  );
};

export default App;
```

配合 Form 使用

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

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: string) => {
    const mentions = getMentions(value);

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

  return (
    <Form form={form} layout="horizontal" onFinish={onFinish}>
      <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',
        },
      ]}
    />
  </Form.Item>
  <Form.Item
    name="bio"
    label="Bio"
    labelCol={{ span: 6 }}
    wrapperCol={{ span: 16 }}
    rules={[{ required: true }]}
  >
    <Mentions
      rows={3}
      placeholder="You can use @ to ref user here"
      options={[
        {
          value: 'afc163',
          label: 'afc163',
        },
        {
          value: 'zombieJ',
          label: 'zombieJ',
        },
        {
          value: 'yesmeck',
          label: 'yesmeck',
        },
      ]}
    />
  </Form.Item>
  <Form.Item wrapperCol={{ span: 14, offset: 6 }}>
    <Space wrap>
```

```
          <Button htmlType="submit" type="primary">
            Submit
          </Button>
          <Button htmlType="button" onClick={onReset}>
            Reset
          </Button>
        </Space>
      </Form.Item>
    </Form>
  );
};

export default App;
```

自定义触发字符

```
import React, { useState } from 'react';
import { Mentions } from 'antd';
import type { MentionsProps } 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<PrefixType>('@');

  const onSearch: MentionsProps['onSearch'] = (_, newPrefix) => {
    setPrefix(newPrefix as PrefixType);
  };

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

```
  );
};

export default App;
```

无效或只读

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

const options = ['afc163', 'zombiej', 'yesmeck'].map((value) => ({
  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;
```

向上展开

```
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;
```

带移除图标

```
import React, { useState } from 'react';
import { CloseSquareFilled } from '@ant-design/icons';
import { Mentions } from 'antd';

const App: React.FC = () => {
  const [value, setValue] = useState('hello world');
  return (
    <>
      <Mentions value={value} onChange={setValue} allowClear />
      <br />
      <br />
      <Mentions
        value={value}
        onChange={setValue}
        allowClear={{ clearIcon: <CloseSquareFilled /> }}
      />
      <br />
      <br />
      <Mentions value={value} onChange={setValue} allowClear rows={3} />
    </>
  );
};
```

```
export default App;
```

自动大小

```
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;
```

自定义状态

```
import React from 'react';
import { Mentions, Space } from 'antd';
import type { GetProp, MentionProps } from 'antd';

type MentionsOptionProps = GetProp<MentionProps, 'options'>[number];

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;
```

_InternalPanelDoNotUseOrYouWillBeFired

Debug

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

```
const { _InternalPanelDoNotUseOrYouWillBeFired: InternalMentions } =
Mentions;

const options = [
  {
    value: 'afc163',
    label: 'afc163',
  },
  {
    value: 'zombieJ',
    label: 'zombieJ',
  },
];

const App: React.FC = () => (
  <InternalMentions style={{ width: '100%' }} value="@" options={options}
/>
);

export default App;
```

组件 Token

Debug

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

const { _InternalPanelDoNotUseOrYouWillBeFired: InternalMentions } =
Mentions;

const options = [
  {
    value: 'afc163',
    label: 'afc163',
  },
  {
    value: 'zombieJ',
    label: 'zombieJ',
  },
];

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

```
      components: { Mentions: { dropdownHeight: 500, controlItemWidth: 300,
zIndexPopup: 1000 } },
    }}
  >
    <InternalMentions style={{ width: '100%' }} value="@" options={options}
/>
  </ConfigProvider>
);

export default App;
```

API

通用属性参考：通用属性

Mentions

参数	说明	类型	默认值	版本
allowClear	可以点击清除图标删除内容	boolean | { clearIcon?: ReactNode }	false	5.13.0
autoFocus	自动获得焦点	boolean	false	
autoSize	自适应内容高度，可设置为 true | false 或对象：{ minRows: 2, maxRows: 6 }	boolean | object	false	
defaultValue	默认值	string	-	
filterOption	自定义过滤逻辑	false | (input: string, option: OptionProps) => boolean	-	
getPopupContainer	指定建议框挂载的 HTML 节点	() => HTMLElement	-	
notFoundContent	当下拉列表为空时显示的内容	ReactNode	Not Found	
placement	弹出层展示位置	top | bottom	bottom	
prefix	设置触发关键字	string | string[]	@	

split	设置选中项前后分隔符	string		
status	设置校验状态	'error' | 'warning'	-	4.19.0
validateSearch	自定义触发验证逻辑	(text: string, props: MentionsProps) => void	-	
value	设置值	string	-	
variant	形态变体	`outlined` | `borderless` | `filled` | `underlined`	`outlined`	5.13.0 | `underlined`: 5.24.0
onBlur	失去焦点时触发	() => void	-	
onChange	值改变时触发	(text: string) => void	-	
onClear	按下清除按钮的回调	() => void	-	5.20.0
onFocus	获得焦点时触发	() => void	-	
onResize	resize 回调	function({ width, height })	-	
onSearch	搜索时触发	(text: string, prefix: string) => void	-	
onSelect	选择选项时触发	(option: OptionProps, prefix: string) => void	-	
onPopupScroll	滚动时触发	(event: Event) => void	-	5.23.0
options	选项配置	Options	[]	5.1.0

Mentions 方法

名称	描述
blur()	移除焦点
focus()	获取焦点

Option

参数	说明	类型	默认值
value	选择时填充的值	string	-
label	选项的标题	React.ReactNode	-
key	选项的 key 值	string	-
disabled	是否可选	boolean	-
className	css 类名	string	-
style	选项样式	React.CSSProperties	-

主题变量（Design Token）