何时使用 {#when-to-use}

- 提供可消费 React context 的 `message.xxx` 、`Modal.xxx` 、`notification.xxx` 的静态方法，可以简化 useMessage 等方法需要手动植入 `contextHolder` 的问题。
- 提供基于 `.ant-app` 的默认重置样式，解决原生元素没有 antd 规范样式的问题。

代码演示

基本用法

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

// Sub page
const MyPage = () => {
  const { message, modal, notification } = App.useApp();

  const showMessage = () => {
    message.success('Success!');
  };

  const showModal = () => {
    modal.warning({
      title: 'This is a warning message',
      content: 'some messages...some messages...',
    });
  };

  const showNotification = () => {
    notification.info({
      message: 'Notification topLeft',
      description: 'Hello, Ant Design!!',
      placement: 'topLeft',
    });
  };

  return (
    <Space wrap>
      <Button type="primary" onClick={showMessage}>
        Open message
      </Button>
      <Button type="primary" onClick={showModal}>
        Open modal
      </Button>
      <Button type="primary" onClick={showNotification}>
        Open notification
      </Button>
```

```
    </Space>
  );
};

// Entry component
export default () => (
  <App>
    <MyPage />
  </App>
);
```

Hooks 配置

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

// Sub page
const MyPage = () => {
  const { message, notification } = App.useApp();

  const showMessage = () => {
    message.success('Success!');
  };

  const showNotification = () => {
    notification.info({
      message: 'Notification',
      description: 'Hello, Ant Design!!',
    });
  };

  return (
    <Space wrap>
      <Button type="primary" onClick={showMessage}>
        Message for only one
      </Button>
      <Button type="primary" onClick={showNotification}>
        Notification for bottomLeft
      </Button>
    </Space>
  );
};

// Entry component
export default () => (
  <App message={{ maxCount: 1 }} notification={{ placement: 'bottomLeft'
```

```
}}>
    <MyPage />
  </App>
);
```

如何使用

基础用法

App 组件通过 `Context` 提供上下文方法调用，因而 useApp 需要作为子组件才能使用，我们推荐在应用中顶层包裹 App。

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

const MyPage: React.FC = () => {
  const { message, notification, modal } = App.useApp();
  message.success('Good!');
  notification.info({ message: 'Good' });
  modal.warning({ title: 'Good' });
  // ....
  // other message, notification, modal static function
  return <div>Hello word</div>;
};

const MyApp: React.FC = () => (
  <App>
    <MyPage />
  </App>
);

export default MyApp;
```

注意：App.useApp 必须在 App 之下方可使用。

与 ConfigProvider 先后顺序

App 组件只能在 `ConfigProvider` 之下才能使用 Design Token， 如果需要使用其样式重置能力，则 ConfigProvider 与 App 组件必须成对出现。

```
<ConfigProvider theme={{ ... }}>
  <App>
    ...
  </App>
</ConfigProvider>
```

内嵌使用场景（如无必要，尽量不做嵌套）

```
<App>
  <Space>
    ...
    <App>...</App>
  </Space>
</App>
```

全局场景（redux 场景）

```
// Entry component
import { App } from 'antd';
import type { MessageInstance } from 'antd/es/message/interface';
import type { ModalStaticFunctions } from 'antd/es/modal/confirm';
import type { NotificationInstance } from 'antd/es/notification/interface';

let message: MessageInstance;
let notification: NotificationInstance;
let modal: Omit<ModalStaticFunctions, 'warn'>;

export default () => {
  const staticFunction = App.useApp();
  message = staticFunction.message;
  modal = staticFunction.modal;
  notification = staticFunction.notification;
  return null;
};

export { message, notification, modal };
```

```
// sub page
import React from 'react';
import { Button, Space } from 'antd';

import { message } from './store';

export default () => {
  const showMessage = () => {
    message.success('Success!');
  };

  return (
    <Space>
```

```
      <Button type="primary" onClick={showMessage}>
        Open message
      </Button>
    </Space>
  );
};
```

API

通用属性参考：通用属性

> *自 `antd@5.1.0` 版本开始提供该组件。*

App

参数	说明	类型	默认值	版本
component	设置渲染元素，为 `false` 则不创建 DOM 节点	ComponentType | false	div	5.11.0
message	App 内 Message 的全局配置	MessageConfig	-	5.3.0
notification	App 内 Notification 的全局配置	NotificationConfig	-	5.3.0

主题变量（Design Token）

FAQ

CSS Var 在 `<App component={false}>` 内不起作用

请确保 App 的 `component` 是一个有效的 html 标签名，以便在启用 CSS 变量时有一个容器来承载 CSS 类名。如果不设置，则默认为 `div` 标签，如果设置为 `false`，则不会创建额外的 DOM 节点，也不会提供默认样式。