全局展示通知提醒信息。

何时使用

在系统四个角显示通知提醒信息。经常用于以下情况：

- 较为复杂的通知内容。
- 带有交互的通知，给出用户下一步的行动点。
- 系统主动推送。

API

- `notification.success(config)`
- `notification.error(config)`
- `notification.info(config)`
- `notification.warning(config)`
- `notification.warn(config)`
- `notification.open(config)`
- `notification.close(key: String)`
- `notification.destroy()`

config 参数如下：

参数	说明	类型	默认值
bottom	消息从底部弹出时，距离底部的位置，单位像素	number	24
btn	自定义关闭按钮	ReactNode	-
className	自定义 CSS class	string	-
closeIcon	自定义关闭图标	ReactNode	-
description	通知提醒内容，必选	ReactNode	-
duration	默认 4.5 秒后自动关闭，配置为 null 则不自动关闭	number	4.5
getContainer	配置渲染节点的输出位置	() => HTMLNode	() => document.body
icon	自定义图标	ReactNode	-
key	当前通知唯一标志	string	-
message	通知提醒标题，必选	ReactNode	-
placement	弹出位置，可选 `topLeft` `topRight` `bottomLeft` `bottomRight`	string	`topRight`
style	自定义内联样式	CSSProperties	-
top	消息从顶部弹出时，距离顶部的位置，单位像素	number	24
onClick	点击通知时触发的回调函数	function	-
onClose	当通知关闭时触发	function	-

还提供了一个全局配置方法，在调用前提前配置，全局一次生效。

- `notification.config(options)`

  > *当你使用 `ConfigProvider` 进行全局化配置时，系统会默认自动开启 RTL 模式。(4.3.0+)*
  >
  > *当你想单独使用，可通过如下设置开启 RTL 模式。*

```
notification.config({
  placement: 'bottomRight',
  bottom: 50,
  duration: 3,
  rtl: true,
});
```

参数	说明	类型	默认值	版本
bottom	消息从底部弹出时，距离底部的位置，单位像素	number	24	
closeIcon	自定义关闭图标	ReactNode	-	
duration	默认自动关闭延时，单位秒	number	4.5	
getContainer	配置渲染节点的输出位置	() => HTMLNode	() => document.body	
placement	弹出位置，可选 `top` `topLeft` `topRight` `bottom` `bottomLeft` `bottomRight`	string	`topRight`	
rtl	是否开启 RTL 模式	boolean	false	
top	消息从顶部弹出时，距离顶部的位置，单位像素	number	24	
maxCount	最大显示数, 超过限制时，最早的消息会被自动关闭	number	-	4.17.0

FAQ

为什么 notification 不能获取 context、redux 的内容和 ConfigProvider 的 `locale/prefixCls` 配置?

直接调用 notification 方法，antd 会通过 `ReactDOM.render` 动态创建新的 React 实体。其 context 与当前代码所在 context 并不相同，因而无法获取 context 信息。

当你需要 context 信息（例如 ConfigProvider 配置的内容）时，可以通过 `notification.useNotification` 方法会返回 `api` 实体以及 `contextHolder` 节点。将其插入到你需要获取 context 位置即可：

```
const [api, contextHolder] = notification.useNotification();

return (
```

```
  <Context1.Provider value="Ant">
    {/* contextHolder 在 Context1 内，它可以获得 Context1 的 context */}
    {contextHolder}
    <Context2.Provider value="Design">
      {/* contextHolder 在 Context2 外，因而不会获得 Context2 的 context */}
    </Context2.Provider>
  </Context1.Provider>
);
```

异同：通过 hooks 创建的 `contextHolder` 必须插入到子元素节点中才会生效，当你不需要上下文信息时请直接调用。

静态方法如何设置 prefixCls？

你可以通过 `ConfigProvider.config` 进行设置。