何时使用 {#when-to-use}

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。

代码演示

Hooks 调用（推荐）

```
import React, { useMemo } from 'react';
import {
  RadiusBottomleftOutlined,
  RadiusBottomrightOutlined,
  RadiusUpleftOutlined,
  RadiusUprightOutlined,
} from '@ant-design/icons';
import { Button, Divider, notification, Space } from 'antd';
import type { NotificationArgsProps } from 'antd';

type NotificationPlacement = NotificationArgsProps['placement'];

const Context = React.createContext({ name: 'Default' });

const App: React.FC = () => {
  const [api, contextHolder] = notification.useNotification();

  const openNotification = (placement: NotificationPlacement) => {
    api.info({
      message: `Notification ${placement}`,
      description: <Context.Consumer>{({ name }) => `Hello, ${name}!`}
</Context.Consumer>,
      placement,
    });
  };

  const contextValue = useMemo(() => ({ name: 'Ant Design' }), []);

  return (
    <Context.Provider value={contextValue}>
      {contextHolder}
      <Space>
        <Button
          type="primary"
          onClick={() => openNotification('topLeft')}
```

```
          icon={<RadiusUpleftOutlined />}
        >
          topLeft
        </Button>
        <Button
          type="primary"
          onClick={() => openNotification('topRight')}
          icon={<RadiusUprightOutlined />}
        >
          topRight
        </Button>
      </Space>
      <Divider />
      <Space>
        <Button
          type="primary"
          onClick={() => openNotification('bottomLeft')}
          icon={<RadiusBottomleftOutlined />}
        >
          bottomLeft
        </Button>
        <Button
          type="primary"
          onClick={() => openNotification('bottomRight')}
          icon={<RadiusBottomrightOutlined />}
        >
          bottomRight
        </Button>
      </Space>
    </Context.Provider>
  );
};

export default App;
```

自动关闭的延时

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

const App: React.FC = () => {
  const [api, contextHolder] = notification.useNotification();

  const openNotification = () => {
    api.open({
      message: 'Notification Title',
```

```
      description:
        'I will never close automatically. This is a purposely very very
long description that has many many characters and words.',
      duration: 0,
    });
  };

  return (
    <>
      {contextHolder}
      <Button type="primary" onClick={openNotification}>
        Open the notification box
      </Button>
    </>
  );
};

export default App;
```

带有图标的通知提醒框

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

type NotificationType = 'success' | 'info' | 'warning' | 'error';

const App: React.FC = () => {
  const [api, contextHolder] = notification.useNotification();

  const openNotificationWithIcon = (type: NotificationType) => {
    api[type]({
      message: 'Notification Title',
      description:
        'This is the content of the notification. This is the content of
the notification. This is the content of the notification.',
    });
  };

  return (
    <>
      {contextHolder}
      <Space>
        <Button onClick={() =>
openNotificationWithIcon('success')}>Success</Button>
        <Button onClick={() =>
openNotificationWithIcon('info')}>Info</Button>
```

```
        <Button onClick={() =>
openNotificationWithIcon('warning')}>Warning</Button>
        <Button onClick={() =>
openNotificationWithIcon('error')}>Error</Button>
      </Space>
    </>
  );
};

export default App;
```

自定义按钮

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

const close = () => {
  console.log(
    'Notification was closed. Either the close button was clicked or
duration time elapsed.',
  );
};

const App: React.FC = () => {
  const [api, contextHolder] = notification.useNotification();

  const openNotification = () => {
    const key = `open${Date.now()}`;
    const btn = (
      <Space>
        <Button type="link" size="small" onClick={() => api.destroy()}>
          Destroy All
        </Button>
        <Button type="primary" size="small" onClick={() =>
api.destroy(key)}>
          Confirm
        </Button>
      </Space>
    );
    api.open({
      message: 'Notification Title',
      description:
        'A function will be be called after the notification is closed
(automatically after the "duration" time of manually).',
      btn,
      key,
```

```
      onClose: close,
    });
  };

  return (
    <>
      {contextHolder}
      <Button type="primary" onClick={openNotification}>
        Open the notification box
      </Button>
    </>
  );
};

export default App;
```

自定义图标

```
import React from 'react';
import { SmileOutlined } from '@ant-design/icons';
import { Button, notification } from 'antd';

const App: React.FC = () => {
  const [api, contextHolder] = notification.useNotification();

  const openNotification = () => {
    api.open({
      message: 'Notification Title',
      description:
        'This is the content of the notification. This is the content of
the notification. This is the content of the notification.',
      icon: <SmileOutlined style={{ color: '#108ee9' }} />,
    });
  };

  return (
    <>
      {contextHolder}
      <Button type="primary" onClick={openNotification}>
        Open the notification box
      </Button>
    </>
  );
};
```

```
export default App;
```

位置

```
import React from 'react';
import {
  BorderBottomOutlined,
  BorderTopOutlined,
  RadiusBottomleftOutlined,
  RadiusBottomrightOutlined,
  RadiusUpleftOutlined,
  RadiusUprightOutlined,
} from '@ant-design/icons';
import { Button, Divider, notification, Space } from 'antd';
import type { NotificationArgsProps } from 'antd';

type NotificationPlacement = NotificationArgsProps['placement'];

const App: React.FC = () => {
  const [api, contextHolder] = notification.useNotification();

  const openNotification = (placement: NotificationPlacement) => {
    api.info({
      message: `Notification ${placement}`,
      description:
        'This is the content of the notification. This is the content of
the notification. This is the content of the notification.',
      placement,
    });
  };

  return (
    <>
      {contextHolder}
      <Space>
        <Button type="primary" onClick={() => openNotification('top')}
icon={<BorderTopOutlined />}>
          top
        </Button>
        <Button
          type="primary"
          onClick={() => openNotification('bottom')}
          icon={<BorderBottomOutlined />}
        >
          bottom
```

```
        </Button>
      </Space>
      <Divider />
      <Space>
        <Button
          type="primary"
          onClick={() => openNotification('topLeft')}
          icon={<RadiusUpleftOutlined />}
        >
          topLeft
        </Button>
        <Button
          type="primary"
          onClick={() => openNotification('topRight')}
          icon={<RadiusUprightOutlined />}
        >
          topRight
        </Button>
      </Space>
      <Divider />
      <Space>
        <Button
          type="primary"
          onClick={() => openNotification('bottomLeft')}
          icon={<RadiusBottomleftOutlined />}
        >
          bottomLeft
        </Button>
        <Button
          type="primary"
          onClick={() => openNotification('bottomRight')}
          icon={<RadiusBottomrightOutlined />}
        >
          bottomRight
        </Button>
      </Space>
    </>
  );
};

export default App;
```

自定义样式

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

```
const App: React.FC = () => {
  const [api, contextHolder] = notification.useNotification();

  const openNotification = () => {
    api.open({
      message: 'Notification Title',
      description:
        'This is the content of the notification. This is the content of
the notification. This is the content of the notification.',
      className: 'custom-class',
      style: {
        width: 600,
      },
    });
  };

  return (
    <>
      {contextHolder}
      <Button type="primary" onClick={openNotification}>
        Open the notification box
      </Button>
    </>
  );
};
export default App;
```

更新消息内容

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

const key = 'updatable';

const App: React.FC = () => {
  const [api, contextHolder] = notification.useNotification();
  const openNotification = () => {
    api.open({
      key,
      message: 'Notification Title',
      description: 'description.',
    });

    setTimeout(() => {
      api.open({
```

```
        key,
        message: 'New Title',
        description: 'New description.',
      });
    }, 1000);
  };

  return (
    <>
      {contextHolder}
      <Button type="primary" onClick={openNotification}>
        Open the notification box
      </Button>
    </>
  );
};

export default App;
```

堆叠

v5.10.0

```
import React, { useMemo } from 'react';
import { Button, Divider, InputNumber, notification, Space, Switch } from
'antd';

const Context = React.createContext({ name: 'Default' });

const App: React.FC = () => {
  const [enabled, setEnabled] = React.useState(true);
  const [threshold, setThreshold] = React.useState(3);
  const [api, contextHolder] = notification.useNotification({
    stack: enabled
      ? {
          threshold,
        }
      : false,
  });

  const openNotification = () => {
    api.open({
      message: 'Notification Title',
      description: `${Array.from(
        { length: Math.round(Math.random() * 5) + 1 },
        () => 'This is the content of the notification.',
```

```
      ).join('\n')}`,
      duration: null,
    });
  };

  const contextValue = useMemo(() => ({ name: 'Ant Design' }), []);

  return (
    <Context.Provider value={contextValue}>
      {contextHolder}
      <div>
        <Space size="large">
          <Space style={{ width: '100%' }}>
            <span>Enabled: </span>
            <Switch checked={enabled} onChange={(v) => setEnabled(v)} />
          </Space>
          <Space style={{ width: '100%' }}>
            <span>Threshold: </span>
            <InputNumber
              disabled={!enabled}
              value={threshold}
              step={1}
              min={1}
              max={10}
              onChange={(v) => setThreshold(v || 0)}
            />
          </Space>
        </Space>
        <Divider />
        <Button type="primary" onClick={openNotification}>
          Open the notification box
        </Button>
      </div>
    </Context.Provider>
  );
};

export default App;
```

显示进度条

v5.18.0

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

```
const App: React.FC = () => {
  const [api, contextHolder] = notification.useNotification();

  const openNotification = (pauseOnHover: boolean) => () => {
    api.open({
      message: 'Notification Title',
      description:
        'This is the content of the notification. This is the content of
the notification. This is the content of the notification.',
      showProgress: true,
      pauseOnHover,
    });
  };

  return (
    <>
      {contextHolder}
      <Space>
        <Button type="primary" onClick={openNotification(true)}>
          Pause on hover
        </Button>
        <Button type="primary" onClick={openNotification(false)}>
          Don&apos;t pause on hover
        </Button>
      </Space>
    </>
  );
};

export default App;
```

静态方法（不推荐）

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

const openNotification = () => {
  notification.open({
    message: 'Notification Title',
    description:
      'This is the content of the notification. This is the content of the
notification. This is the content of the notification.',
    onClick: () => {
      console.log('Notification Clicked!');
    },
  });
```

```
};
const App: React.FC = () => (
  <Button type="primary" onClick={openNotification}>
    Open the notification box
  </Button>
);

export default App;
```

_InternalPanelDoNotUseOrYouWillBeFired

Debug

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

/** Test usage. Do not use in your production. */
const { _InternalPanelDoNotUseOrYouWillBeFired: InternalPanel } =
notification;

export default () => (
  <InternalPanel
    message="Hello World!"
    description="Hello World?"
    type="success"
    actions={
      <Button type="primary" size="small">
        My Button
      </Button>
    }
  />
);
```

API

通用属性参考：通用属性

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

config 参数如下：

参数	说明	类型	
actions	自定义按钮组	ReactNode	-
~~btn~~	自定义按钮组，请使用 `actions` 替换	ReactNode	-
className	自定义 CSS class	string	-
closeIcon	自定义关闭图标	ReactNode	tr
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.
showProgress	显示自动关闭通知框的进度条	boolean	
pauseOnHover	悬停时是否暂停计时器	boolean	tr
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 `top` | `topLeft` | `topRight` | `bottom` | `bottomLeft` | `bottomRight`	string	to
style	自定义内联样式	CSSProperties	-
role	供屏幕阅读器识别的通知内容语义，默认为 `alert`。此情况下屏幕阅读器会立即打断当前正在阅读的其他内容，转而阅读通知内容	`alert` | `status`	al
onClick	点击通知时触发的回调函数	function	-
onClose	当通知关闭时触发	function	-
props	透传至通知 `div` 上的 props 对象，支持传入 `data-*` `aria-*` 或 `role` 作为对象的属性。需要注意的是，虽然在 TypeScript 类型中声明的类型支持传入 `data-*` 作为对象的属性，但目前只允许传入 `data-testid` 作为对象的属性。 详见 https://github.com/microsoft/TypeScript/issues/28960	Object	-

- `notification.useNotification(config)`

config 参数如下：

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	true	5.7.0：设置为 null 或 false 时隐藏关闭按钮
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`	
showProgress	显示自动关闭通知框的进度条	boolean		5.18.0
pauseOnHover	悬停时是否暂停计时器	boolean	true	5.18.0
rtl	是否开启 RTL 模式	boolean	false	
stack	堆叠模式，超过阈值时会将所有消息收起	boolean | `{ threshold: number }`	`{ threshold: 3 }`	5.10.0
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

全局配置

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,
});
```

notification.config

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	true	5.7.0：设置为 null 或 false 时隐藏关闭按钮
duration	默认自动关闭延时，单位秒	number	4.5	
showProgress	显示自动关闭通知框的进度条	boolean		5.18.0
pauseOnHover	悬停时是否暂停计时器	boolean	true	5.18.0
getContainer	配置渲染节点的输出位置，但依旧为全屏展示	() => 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

主题变量（Design Token）

FAQ

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

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。

> *可通过 App 包裹组件 简化 `useNotification` 等方法需要手动植入 contextHolder 的问题。*

静态方法如何设置 prefixCls？

你可以通过 `ConfigProvider.config` 进行设置。