何时使用 {#when-to-use}

- 页面需要添加水印标识版权时使用。
- 适用于防止信息盗用。

代码演示

基本

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

const App: React.FC = () => (
  <Watermark content="Ant Design">
    <div style={{ height: 500 }} />
  </Watermark>
);

export default App;
```

多行水印

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

const App: React.FC = () => (
  <Watermark content={['Ant Design', 'Happy Working']}>
    <div style={{ height: 500 }} />
  </Watermark>
);

export default App;
```

图片水印

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

const App: React.FC = () => (
  <Watermark
    height={30}
    width={130}

image="https://mdn.alipayobjects.com/huamei_7uahnr/afts/img/A*lkAoRbywo0oAAAA
  >
```

```
    <div style={{ height: 500 }} />
  </Watermark>
);

export default App;
```

自定义配置

```
import React, { useState } from 'react';
import { ColorPicker, Flex, Form, Input, InputNumber, Slider, Typography,
Watermark } from 'antd';
import type { ColorPickerProps, GetProp, WatermarkProps } from 'antd';

type Color = Extract<GetProp<ColorPickerProps, 'value'>, string | {
cleared: any }>;

const { Paragraph } = Typography;

interface WatermarkConfig {
  content: string;
  color: string | Color;
  fontSize: number;
  zIndex: number;
  rotate: number;
  gap: [number, number];
  offset?: [number, number];
}

const App: React.FC = () => {
  const [form] = Form.useForm();
  const [config, setConfig] = useState<WatermarkConfig>({
    content: 'Ant Design',
    color: 'rgba(0, 0, 0, 0.15)',
    fontSize: 16,
    zIndex: 11,
    rotate: -22,
    gap: [100, 100],
    offset: undefined,
  });
  const { content, color, fontSize, zIndex, rotate, gap, offset } = config;

  const watermarkProps: WatermarkProps = {
    content,
    zIndex,
    rotate,
    gap,
```

```
    offset,
    font: { color: typeof color === 'string' ? color : color.toRgbString(),
fontSize },
  };

  return (
    <Flex gap="middle">
      <Watermark {...watermarkProps}>
        <Typography>
          <Paragraph>
            The light-speed iteration of the digital world makes products
more complex. However,
            human consciousness and attention resources are limited. Facing
this design
            contradiction, the pursuit of natural interaction will be the
consistent direction of
            Ant Design.
          </Paragraph>
          <Paragraph>
            Natural user cognition: According to cognitive psychology,
about 80% of external
            information is obtained through visual channels. The most
important visual elements in
            the interface design, including layout, colors, illustrations,
icons, etc., should fully
            absorb the laws of nature, thereby reducing the user&amp;apos;s
cognitive cost and bringing
            authentic and smooth feelings. In some scenarios, opportunely
adding other sensory
            channels such as hearing, touch can create a richer and more
natural product experience.
          </Paragraph>
          <Paragraph>
            Natural user behavior: In the interaction with the system, the
designer should fully
            understand the relationship between users, system roles, and
task objectives, and also
            contextually organize system functions and services. At the
same time, a series of
            methods such as behavior analysis, artificial intelligence and
sensors could be applied
            to assist users to make effective decisions and reduce extra
operations of users, to
            save users&amp;apos; mental and physical resources and make human-
computer interaction more
            natural.
```

```
          </Paragraph>
        </Typography>
        <img
          style={{ zIndex: 10, width: '100%', maxWidth: 800, position:
'relative' }}

src="https://gw.alipayobjects.com/mdn/rms_08e378/afts/img/A*zx7LTI_ECSAAAAAAA
          alt="img"
        />
      </Watermark>
      <Form
        style={{ width: 280, flexShrink: 0, borderLeft: '1px solid #eee',
paddingInlineStart: 16 }}
        form={form}
        layout="vertical"
        initialValues={config}
        onValuesChange={(_, values) => {
          setConfig(values);
        }}
      >
        <Form.Item name="content" label="Content">
          <Input placeholder="请输入" />
        </Form.Item>
        <Form.Item name="color" label="Color">
          <ColorPicker />
        </Form.Item>
        <Form.Item name="fontSize" label="FontSize">
          <Slider step={1} min={1} max={100} />
        </Form.Item>
        <Form.Item name="zIndex" label="zIndex">
          <Slider step={1} min={0} max={100} />
        </Form.Item>
        <Form.Item name="rotate" label="Rotate">
          <Slider step={1} min={-180} max={180} />
        </Form.Item>
        <Form.Item label="Gap" style={{ marginBottom: 0 }}>
          <Flex gap="small">
            <Form.Item name={['gap', 0]}>
              <InputNumber placeholder="gapX" style={{ width: '100%' }} />
            </Form.Item>
            <Form.Item name={['gap', 1]}>
              <InputNumber placeholder="gapY" style={{ width: '100%' }} />
            </Form.Item>
          </Flex>
        </Form.Item>
        <Form.Item label="Offset" style={{ marginBottom: 0 }}>
```

```
          <Flex gap="small">
            <Form.Item name={['offset', 0]}>
              <InputNumber placeholder="offsetLeft" style={{ width: '100%'
}} />
            </Form.Item>
            <Form.Item name={['offset', 1]}>
              <InputNumber placeholder="offsetTop" style={{ width: '100%'
}} />
            </Form.Item>
          </Flex>
        </Form.Item>
      </Form>
    </Flex>
  );
};

export default App;
```

Modal 与 Drawer

```
import React from 'react';
import { Button, Drawer, Flex, Modal, Watermark } from 'antd';

const style: React.CSSProperties = {
  height: 300,
  display: 'flex',
  justifyContent: 'center',
  alignItems: 'center',
  backgroundColor: 'rgba(150, 150, 150, 0.2)',
};

const placeholder = <div style={style}>A mock height</div>;

const App: React.FC = () => {
  const [showModal, setShowModal] = React.useState(false);
  const [showDrawer, setShowDrawer] = React.useState(false);
  const [showDrawer2, setShowDrawer2] = React.useState(false);

  const closeModal = () => setShowModal(false);
  const closeDrawer = () => setShowDrawer(false);
  const closeDrawer2 = () => setShowDrawer2(false);

  return (
    <>
      <Flex gap="middle">
        <Button type="primary" onClick={() => setShowModal(true)}>
```

```
          Show in Modal
        </Button>
        <Button type="primary" onClick={() => setShowDrawer(true)}>
          Show in Drawer
        </Button>
        <Button type="primary" onClick={() => setShowDrawer2(true)}>
          Not Show in Drawer
        </Button>
      </Flex>
      <Watermark content="Ant Design">
        <Modal
          destroyOnClose
          open={showModal}
          title="Modal"
          onCancel={closeModal}
          onOk={closeModal}
        >
          {placeholder}
        </Modal>
        <Drawer destroyOnClose open={showDrawer} title="Drawer" onClose=
{closeDrawer}>
          {placeholder}
        </Drawer>
      </Watermark>
      <Watermark content="Ant Design" inherit={false}>
        <Drawer destroyOnClose open={showDrawer2} title="Drawer" onClose=
{closeDrawer2}>
          {placeholder}
        </Drawer>
      </Watermark>
    </>
  );
};

export default App;
```

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。*

Watermark

参数	说明	类型	默认值	版本

width	水印的宽度，`content` 的默认值为自身的宽度	number	120	
height	水印的高度，`content` 的默认值为自身的高度	number	64	
inherit	是否将水印传导给弹出组件如 Modal、Drawer	boolean	true	5.11.0
rotate	水印绘制时，旋转的角度，单位 °	number	-22	
zIndex	追加的水印元素的 z-index	number	9	
image	图片源，建议导出 2 倍或 3 倍图，优先级高 (支持 base64 格式)	string	-	
content	水印文字内容	string | string[]	-	
font	文字样式	Font	Font	
gap	水印之间的间距	[number, number]	[100, 100]	
offset	水印距离容器左上角的偏移量，默认为 `gap/2`	[number, number]	[gap[0]/2, gap[1]/2]	

Font

参数	说明	类型	默认值	版本
color	字体颜色	CanvasFillStrokeStyles.fillStyle	rgba(0,0,0,.15)	
fontSize	字体大小	number	16	
fontWeight	字体粗细	`normal` | `light` | `weight` | number	normal	
fontFamily	字体类型	string	sans-serif	
fontStyle	字体样式	`none` | `normal` | `italic` | `oblique`	normal	
textAlign	指定文本对齐方向	CanvasTextAlign	`center`	5.10.0

主题变量（Design Token）

FAQ

处理异常图片水印

当使用图片水印且图片加载异常时，可以同时添加 `content` 防止水印失效（自 5.2.3 开始支持）。

```
<Watermark
  height={30}
  width={130}
  content="Ant Design"

image="https://mdn.alipayobjects.com/huamei_7uahnr/afts/img/A*lkAoRbywo0oAAAA
>
  <div style={{ height: 500 }} />
</Watermark>
```

从 5.18.0 版本后，为什么添加了 `overflow: hidden` 样式?

在之前版本，用户可以通过开发者工具将容器高度设置为 0 来隐藏水印，为了避免这种情况，我们在容器上添加了 `overflow: hidden` 样式。当容器高度变化时，则内容也一同被隐藏。你可以通过覆盖样式来修改这个行为：

```
<Watermark style={{ overflow: 'visible' }} />
```