


使用方法

使用图标组件，你需要安装 [@ant-design/icons](#) 图标组件包：

设计师专属

安装 [Kitchen Sketch 插件](#) ，就可以一键拖拽使用 Ant Design 和 Iconfont 的海量图标，还可以关联自有项目。

图标列表

代码演示

基本用法

```
import React from 'react';
import {
  HomeOutlined,
  LoadingOutlined,
  SettingFilled,
  SmileOutlined,
  SyncOutlined,
} from '@ant-design/icons';
import { Space } from 'antd';

const App: React.FC = () => (
  <Space>
    <HomeOutlined />
    <SettingFilled />
    <SmileOutlined />
    <SyncOutlined spin />
    <SmileOutlined rotate={180} />
    <LoadingOutlined />
  </Space>
);

export default App;
```

多色图标

```
import React from 'react';
import { CheckCircleTwoTone, HeartTwoTone, SmileTwoTone } from '@ant-
design/icons';
import { Space } from 'antd';
```

```

const App: React.FC = () => (
  <Space>
    <SmileTwoTone />
    <HeartTwoTone twoToneColor="#eb2f96" />
    <CheckCircleTwoTone twoToneColor="#52c41a" />
  </Space>
);

export default App;

```

自定义图标

```

import React from 'react';
import Icon, { HomeOutlined } from '@ant-design/icons';
import { Space } from 'antd';
import type { GetProps } from 'antd';

type CustomIconComponentProps = GetProps<typeof Icon>;

const HeartSvg = () => (
  <svg width="1em" height="1em" fill="currentColor" viewBox="0 0 1024 1024">
    <title>heart icon</title>
    <path d="M923 283.6c-13.4-31.1-32.6-58.9-56.9-82.8-24.3-23.8-52.5-42.4-84-55.5-32.5-13.5-66.9-20.3-102.4-20.3-49.3 0-97.4 13.5-139.2 39-10 6.1-19.5 12.8-28.5 20.1-9-7.3-18.5-14-28.5-20.1-41.8-25.5-89.9-39-139.2-39-35.5 0-69.9 6.8-102.4 20.3-31.4 13-59.7 31.7-84 55.5-24.4 23.9-43.5 51.7-56.9 82.8-13.9 32.3-21 66.6-21 101.9 0 33.3 6.8 68 20.3 103.3 11.3 29.5 27.5 60.1 48.2 91 32.8 48.9 77.9 99.9 133.9 151.6 92.8 85.7 184.7 144.9 188.6 147.3l23.7 15.2c10.5 6.7 24 6.7 34.5 0l23.7-15.2c3.9-2.5 95.7-61.6 188.6-147.3 56-51.7 101.1-102.7 133.9-151.6 20.7-30.9 37-61.5 48.2-91 13.5-35.3 20.3-70 20.3-103.3 0.1-35.3-7-69.6-20.9-101.9z" />
  </svg>
);

const PandaSvg = () => (
  <svg viewBox="0 0 1024 1024" width="1em" height="1em" fill="currentColor">
    <title>Panda icon</title>
    <path
      d="M99.096 315.634s-82.58-64.032-82.58-132.13c0-66.064 33.032-165.162 148.646-148.646 83.37 11.91 99.096 165.162 99.096 165.162l-165.162 115.614zM924.906 315.634s82.58-64.032 82.58-132.13c0-66.064-33.032-165.162-148.646-148.646-83.37 11.91-99.096 165.162-99.096 165.162l165.162 115.614z"
      fill="#6B676E"
    />
  </svg>
);

```

```

    <path
      d="M1024 561.548c0 264.526-229.23 429.42-512.002 429.4250 826.076 0
561.548 283.96 66.064 512.002 66.064 1024 297.022 1024 561.548z"
      fill="#FFEBD2"
    />
    <path
      d="M330.324 842.126c0 82.096 81.34 148.646 181.678 148.646s181.678-
66.55 181.678-148.646H330.324z"
      fill="#E9D7C3"
    />
    <path
      d="M644.13 611.098C594.582 528.516 561.55 512 512.002 512c-49.548 0-
82.58 16.516-132.13 99.096-42.488 70.814-78.73 211.264-49.548 247.742
66.064 82.58 165.162 33.032 181.678 33.032 16.516 0 115.614 49.548 181.678-
33.032 29.18-36.476-7.064-176.93-49.55-247.74z"
      fill="#FFFFFF"
    />
    <path
      d="M611.098 495.484c0-45.608 36.974-82.58 82.58-82.58 49.548 0
198.194 99.098 198.194 165.162s-79.934 144.904-148.646 99.096c-49.548-
33.032-132.128-148.646-132.128-181.678zM412.904 495.484c0-45.608-36.974-
82.58-82.58-82.58-49.548 0-198.194 99.098-198.194 165.162s79.934 144.904
148.646 99.096c49.548-33.032 132.128-148.646 132.128-181.678z"
      fill="#6B676E"
    />
    <path
      d="M512.002 726.622c-30.06 0-115.614 5.668-115.614 33.032 0 49.638
105.484 85.24 115.614 82.58 10.128 2.66 115.614-32.944 115.614-82.58-0.002-
27.366-85.556-33.032-115.614-33.032z"
      fill="#464655"
    />
    <path
      d="M330.324 495.484m-33.032 0a33.032 33.032 0 1 0 66.064 0 33.032
33.032 0 1 0-66.064 0Z"
      fill="#464655"
    />
    <path
      d="M693.678 495.484m-33.032 0a33.032 33.032 0 1 0 66.064 0 33.032
33.032 0 1 0-66.064 0Z"
      fill="#464655"
    />
  </svg>
);

```

```

const HeartIcon = (props: Partial<CustomIconComponentProps>) => (
  <Icon component={HeartSvg} {...props} />

```

```

);

const PandaIcon = (props: Partial<CustomIconComponentProps>) => (
  <Icon component={PandaSvg} {...props} />
);

const App: React.FC = () => (
  <Space>
    <HeartIcon style={{ color: 'hotpink' }} />
    <PandaIcon style={{ fontSize: '32px' }} />
    <Icon component={HomeOutlined as React.ForwardRefExoticComponent<any>}
  />
  <HomeOutlined />
</Space>
);

export default App;

```

使用 iconfont.cn

```

import React from 'react';
import { createFromIconfontCN } from '@ant-design/icons';
import { Space } from 'antd';

const IconFont = createFromIconfontCN({
  scriptUrl: '//at.alicdn.com/t/font_8d5l8fzk5b87iudi.js',
});

const App: React.FC = () => (
  <Space>
    <IconFont type="icon-tuichu" />
    <IconFont type="icon-facebook" style={{ color: '#1877F2' }} />
    <IconFont type="icon-twitter" />
  </Space>
);

export default App;

```

使用 iconfont.cn 的多个资源

```

import React from 'react';
import { createFromIconfontCN } from '@ant-design/icons';
import { Space } from 'antd';

```

```
const IconFont = createFromIconfontCN({
  scriptUrl: [
    '//at.alicdn.com/t/font_1788044_0dwu4guekcwr.js', // icon-javascript,
    icon-java, icon-shoppingcart (overridden)
    '//at.alicdn.com/t/font_1788592_a5xf2bdic3u.js', // icon-shoppingcart,
    icon-python
  ],
});

const App: React.FC = () => (
  <Space>
    <IconFont type="icon-javascript" />
    <IconFont type="icon-java" />
    <IconFont type="icon-shoppingcart" />
    <IconFont type="icon-python" />
  </Space>
);

export default App;
```

API

从 4.0 开始，antd 不再内置 Icon 组件，请使用独立的包 `@ant-design/icons`。

通用图标

| 参数 | 说明 | 类型 | 默认值 | 版本 |
|--------------|---|-------------------|-------|----|
| className | 设置图标的样式名 | string | - | |
| rotate | 图标旋转角度（IE9 无效） | number | - | |
| spin | 是否有旋转动画 | boolean | false | |
| style | 设置图标的样式，例如 <code>fontSize</code> 和 <code>color</code> | CSSProperties | - | |
| twoToneColor | 仅适用双色图标。设置双色图标的主要颜色，支持设置十六进制颜色字符串 | string string[] | - | |

其中我们提供了三种主题的图标，不同主题的 Icon 组件名为图标名加主题做为后缀。

```
import { StarOutlined, StarFilled, StarTwoTone } from '@ant-design/icons';

<StarOutlined />
<StarFilled />
<StarTwoTone twoToneColor="#eb2f96" />
```

自定义 Icon

| 参数 | 说明 | 类型 | 默认值 | 版本 |
|-----------|--|--|-------|----|
| component | 控制如何渲染图标，通常是一个渲染根标签为 <code><svg></code> 的 React 组件 | <code>ComponentType<CustomIconComponentProps></code> | - | |
| rotate | 图标旋转角度 (IE9 无效) | number | - | |
| spin | 是否有旋转动画 | boolean | false | |
| style | 设置图标的样式，例如 <code>fontSize</code> 和 <code>color</code> | CSSProperties | - | |

关于 SVG 图标

在 3.9.0 之后，我们使用了 SVG 图标替换了原先的 font 图标，从而带来了以下优势：

- 完全离线化使用，不需要从 CDN 下载字体文件，图标不会因为网络问题呈现方块，也无需字体文件本地部署。
- 在低端设备上 SVG 有更好的清晰度。
- 支持多色图标。
- 对于内建图标的更换可以提供更多 API，而不需要进行样式覆盖。

更多讨论可参考：[#10353](#)。

所有的图标都会以 `<svg>` 标签渲染，可以使用 `style` 和 `className` 设置图标的大小和单色图标的颜色。例如：

```
import { MessageOutlined } from '@ant-design/icons';

<MessageOutlined style={{ fontSize: '16px', color: '#08c' }} />;
```

双色图标主色

对于双色图标，可以通过使用 `getTwoToneColor()` 和 `setTwoToneColor(colorString)` 来全局设置图标主色。

```
import { getTwoToneColor, setTwoToneColor } from '@ant-design/icons';
```

```
setTwoToneColor('#eb2f96');
getTwoToneColor(); // #eb2f96
```

自定义 font 图标

在 3.9.0 之后，我们提供了一个 `createFromIconfontCN` 方法，方便开发者调用在 iconfont.cn 上自行管理的图标。

```
import React from 'react';
import { createFromIconfontCN } from '@ant-design/icons';
import ReactDOM from 'react-dom/client';

const MyIcon = createFromIconfontCN({
  scriptUrl: 'https://at.alicdn.com/t/font_8d5l8fzk5b87iudi.js', // 在
  iconfont.cn 上生成
});

ReactDOM.createRoot(mountNode).render(<MyIcon type="icon-example" />);
```

其本质上是创建了一个使用 `<use>` 标签来渲染图标的组件。

options 的配置项如下：

| 参数 | 说明 | 类型 | 默认值 | 版本 |
|------------------|--|------------------------|-----|----|
| extraCommonProps | 给所有的 svg 图标 <code><Icon /></code> 组件设置额外的属性 | { [key: string]: any } | {} | |
| scriptUrl | iconfont.cn 项目在线生成的 js 地址，@ant-design/icons@4.1.0 之后支持 string[] 类型 | string string[] | - | |

在 `scriptUrl` 都设置有效的情况下，组件在渲染前会自动引入 iconfont.cn 项目中的图标符号集，无需手动引入。

见 [iconfont.cn 使用帮助](#) 查看如何生成 js 地址。

自定义 SVG 图标

如果使用 webpack，可以通过配置 [@svgr/webpack](#) 来将 svg 图标作为 React 组件导入。
`@svgr/webpack` 的 options 选项请参阅 [svgr 文档](#)。

```
// webpack.config.js
module.exports = {
  // ... other config
```

```
test: /\.svg(\?v=\d+\.\d+\.\d+)?$/,\nuse: [\n  {\n    loader: 'babel-loader',\n  },\n  {\n    loader: '@svgr/webpack',\n    options: {\n      babel: false,\n      icon: true,\n    },\n  },\n],\n};
```

如果使用 vite，可以通过配置 [vite-plugin-svg](#) 来将 svg 图标作为 React 组件导入。vite-plugin-svg 的 options 选项请参阅 [svgr 文档](#)。

```
// vite.config.js\nexport default defineConfig(() => ({\n  // ... other config\n  plugins: [svgr({ svgrOptions: { icon: true } })],\n}));
```

```
import React from 'react';\nimport Icon from '@ant-design/icons';\nimport MessageSvg from 'path/to/message.svg'; // 你的 '*.svg' 文件路径\n\n// import MessageSvg from 'path/to/message.svg?react'; // 使用vite 你的\n// '*.svg?react' 文件路径.\nimport ReactDOM from 'react-dom/client';\n\n// in create-react-app:\n// import { ReactComponent as MessageSvg } from 'path/to/message.svg';\n\nReactDOM.createRoot(mountNode).render(<Icon component={MessageSvg} />);
```

Icon 中的 component 组件的接受的属性如下：

| 字段 | 说明 | 类型 | 只读值 | 版本 |
|-----------|-------------|-----------------|--------------|----|
| className | 计算后的 svg 类名 | string | - | |
| fill | svg 元素填充的颜色 | string | currentColor | |
| height | svg 元素高度 | string number | 1em | |

| | | | | |
|-------|---------------|-----------------|-----|--|
| style | 计算后的 svg 元素样式 | CSSProperties | - | |
| width | svg 元素宽度 | string number | 1em | |

主题变量 (Design Token)