When To Use

- When a resource needs long time to load.
- When the component contains lots of information, such as List or Card.
- Only works when loading data for the first time.
- Could be replaced by Spin in any situation, but can provide a better user experience.

Examples

Basic

```tsx
import React from 'react';
import { Skeleton } from 'antd';

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

export default App;
```

Complex combination

```tsx
import React from 'react';
import { Skeleton } from 'antd';

const App: React.FC = () => <Skeleton avatar paragraph={{ rows: 4 }} />;

export default App;
```

Active Animation

```tsx
import React from 'react';
import { Skeleton } from 'antd';

const App: React.FC = () => <Skeleton active />;

export default App;
```

Button/Avatar/Input/Image/Node

```tsx
import React, { useState } from 'react';
import { DotChartOutlined } from '@ant-design/icons';
import type { RadioChangeEvent } from 'antd';
import { Flex, Divider, Form, Radio, Skeleton, Space, Switch } from 'antd';

type SizeType = 'default' | 'small' | 'large';
type ButtonShapeType = 'circle' | 'square' | 'round' | 'default';
```

```tsx
type AvatarShapeType = 'circle' | 'square';

const App: React.FC = () => {
  const [active, setActive] = useState(false);
  const [block, setBlock] = useState(false);
  const [size, setSize] = useState<SizeType>('default');
  const [buttonShape, setButtonShape] = useState<ButtonShapeType>
('default');
  const [avatarShape, setAvatarShape] = useState<AvatarShapeType>
('circle');

  const handleActiveChange = (checked: boolean) => {
    setActive(checked);
  };

  const handleBlockChange = (checked: boolean) => {
    setBlock(checked);
  };

  const handleSizeChange = (e: RadioChangeEvent) => {
    setSize(e.target.value);
  };

  const handleShapeButton = (e: RadioChangeEvent) => {
    setButtonShape(e.target.value);
  };

  const handleAvatarShape = (e: RadioChangeEvent) => {
    setAvatarShape(e.target.value);
  };

  return (
    <Flex gap="middle" vertical>
      <Space>
        <Skeleton.Button active={active} size={size} shape={buttonShape}
block={block} />
        <Skeleton.Avatar active={active} size={size} shape={avatarShape} />
        <Skeleton.Input active={active} size={size} />
      </Space>
      <Skeleton.Button active={active} size={size} shape={buttonShape}
block={block} />
      <Skeleton.Input active={active} size={size} block={block} />
      <Space>
        <Skeleton.Image active={active} />
        <Skeleton.Node active={active} style={{ width: 160 }} />
        <Skeleton.Node active={active}>
```

```
          <DotChartOutlined style={{ fontSize: 40, color: '#bfbfbf' }} />
        </Skeleton.Node>
      </Space>
      <Divider />
      <Form layout="inline" style={{ margin: '16px 0' }}>
        <Space size={16} wrap>
          <Form.Item label="Active">
            <Switch checked={active} onChange={handleActiveChange} />
          </Form.Item>
          <Form.Item label="Button and Input Block">
            <Switch checked={block} onChange={handleBlockChange} />
          </Form.Item>
          <Form.Item label="Size">
            <Radio.Group value={size} onChange={handleSizeChange}>
              <Radio.Button value="default">Default</Radio.Button>
              <Radio.Button value="large">Large</Radio.Button>
              <Radio.Button value="small">Small</Radio.Button>
            </Radio.Group>
          </Form.Item>
          <Form.Item label="Button Shape">
            <Radio.Group value={buttonShape} onChange={handleShapeButton}>
              <Radio.Button value="default">Default</Radio.Button>
              <Radio.Button value="square">Square</Radio.Button>
              <Radio.Button value="round">Round</Radio.Button>
              <Radio.Button value="circle">Circle</Radio.Button>
            </Radio.Group>
          </Form.Item>
          <Form.Item label="Avatar Shape">
            <Radio.Group value={avatarShape} onChange={handleAvatarShape}>
              <Radio.Button value="square">Square</Radio.Button>
              <Radio.Button value="circle">Circle</Radio.Button>
            </Radio.Group>
          </Form.Item>
        </Space>
      </Form>
    </Flex>
  );
};

export default App;
```

Contains sub component

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

```tsx
const App: React.FC = () => {
  const [loading, setLoading] = useState<boolean>(false);

  const showSkeleton = () => {
    setLoading(true);
    setTimeout(() => {
      setLoading(false);
    }, 3000);
  };

  return (
    <Space direction="vertical" style={{ width: '100%' }} size={16}>
      <Skeleton loading={loading}>
        <h4 style={{ marginBottom: 16 }}>Ant Design, a design language</h4>
        <p>
          We supply a series of design principles, practical patterns and
high quality design
          resources (Sketch and Axure), to help people create their product
prototypes beautifully
          and efficiently.
        </p>
      </Skeleton>
      <Button onClick={showSkeleton} disabled={loading}>
        Show Skeleton
      </Button>
    </Space>
  );
};

export default App;
```

List

```tsx
import React, { useState } from 'react';
import type Icon from '@ant-design/icons';
import { LikeOutlined, MessageOutlined, StarOutlined } from '@ant-
design/icons';
import { Avatar, List, Skeleton, Switch } from 'antd';

interface IconTextProps {
  icon: typeof Icon;
  text: React.ReactNode;
}

const listData = Array.from({ length: 3 }).map((_, i) => ({
  href: 'https://ant.design',
```

```
  title: `ant design part ${i + 1}`,
  avatar: `https://api.dicebear.com/7.x/miniavs/svg?seed=${i}`,
  description:
    'Ant Design, a design language for background applications, is refined
by Ant UED Team.',
  content:
    'We supply a series of design principles, practical patterns and high
quality design resources (Sketch and Axure), to help people create their
product prototypes beautifully and efficiently.',
}));

const IconText: React.FC<IconTextProps> = ({ icon, text }) => (
  <>
    {React.createElement(icon, { style: { marginInlineEnd: 8 } })}
    {text}
  </>
);

const App: React.FC = () => {
  const [loading, setLoading] = useState(true);

  const onChange = (checked: boolean) => {
    setLoading(!checked);
  };

  return (
    <>
      <Switch checked={!loading} onChange={onChange} style={{ marginBottom:
16 }} />
      <List
        itemLayout="vertical"
        size="large"
        dataSource={listData}
        renderItem={(item) => (
          <List.Item
            key={item.title}
            actions={
              !loading
                ? [
                    <IconText icon={StarOutlined} text="156" key="list-
vertical-star-o" />,
                    <IconText icon={LikeOutlined} text="156" key="list-
vertical-like-o" />,
                    <IconText icon={MessageOutlined} text="2" key="list-
vertical-message" />,
                  ]
```

```
                : undefined
            }
            extra={
              !loading && (
                <img
                  width={272}
                  alt="logo"
src="https://gw.alipayobjects.com/zos/rmsportal/mqaQswcyDLcXyDKnZfES.png"
                />
              )
            }
          >
            <Skeleton loading={loading} active avatar>
              <List.Item.Meta
                avatar={<Avatar src={item.avatar} />}
                title={<a href={item.href}>{item.title}</a>}
                description={item.description}
              />
              {item.content}
            </Skeleton>
          </List.Item>
        )}
      />
    </>
  );
};

export default App;
```

Custom component token

Debug

```
import React from 'react';
import { ConfigProvider, Skeleton } from 'antd';

const App: React.FC = () => (
  <ConfigProvider
    theme={{
      components: {
        Skeleton: {
          blockRadius: 30,
          titleHeight: 50,
          gradientFromColor: '#222',
          gradientToColor: '#444',
```

```
          paragraphMarginTop: 30,
          paragraphLiHeight: 30,
        },
      },
    }}
  >
    <Skeleton loading active />
  </ConfigProvider>
);

export default App;
```

API

Common props ref：Common props

Skeleton

Property	Description	Type	Default
active	Show animation effect	boolean	false
avatar	Show avatar placeholder	boolean | SkeletonAvatarProps	false
loading	Display the skeleton when true	boolean	-
paragraph	Show paragraph placeholder	boolean | SkeletonParagraphProps	true
round	Show paragraph and title radius when true	boolean	false
title	Show title placeholder	boolean | SkeletonTitleProps	true

SkeletonAvatarProps

Property	Description	Type	Default
active	Show animation effect, only valid when used avatar independently	boolean	false
shape	Set the shape of avatar	`circle` | `square`	-
size	Set the size of avatar	number | `large` | `small` | `default`	-

SkeletonTitleProps

Property	Description	Type	Default

width	Set the width of title	number | string	-

SkeletonParagraphProps

Property	Description	Type	Default
rows	Set the row count of paragraph	number	-
width	Set the width of paragraph. When width is an Array, it can set the width of each row. Otherwise only set the last row width	number | string | Array<number | string>	-

SkeletonButtonProps

Property	Description	Type	Default	Version
active	Show animation effect	boolean	false	
block	Option to fit button width to its parent width	boolean	false	4.17.0
shape	Set the shape of button	`circle` | `round` | `square` | `default`	-	
size	Set the size of button	`large` | `small` | `default`	-	

SkeletonInputProps

Property	Description	Type	Default
active	Show animation effect	boolean	false
size	Set the size of input	`large` | `small` | `default`	-

Design Token