

Drawer

A panel which slides in from the edge of the screen.

When To Use

A Drawer is a panel that is typically overlaid on top of a page and slides in from the side. It contains a set of information or actions. Since the user can interact with the Drawer without leaving the current page, tasks can be achieved more efficiently within the same context.

- Use a Form to create or edit a set of information.
- Processing subtasks. When subtasks are too heavy for a Popover and we still want to keep the subtasks in the context of the main task, Drawer comes very handy.
- When the same Form is needed in multiple places.

Examples

Open

Basic [✎](#)

Basic drawer.

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

const App: React.FC = () => {
  const [open, setOpen] = useState(false);

  const showDrawer = () => {
    setOpen(true);
  };

  const onClose = () => {
    setOpen(false);
  };

  return (
    <>
      <Button type="primary" onClick={showDrawer}>
        Open
      </Button>
      <Drawer title="Basic Drawer" placement='
        <p>Some contents...</p>
        <p>Some contents...</p>
        <p>Some contents...</p>
      </Drawer>
    </>
  );
};

export default App;
```

☐ top ☐ right ☐ bottom
☒ left

Open

Custom Placement [✎](#)

The Drawer can appear from any edge of the screen.

```
import React, { useState } from 'react';
import type { DrawerProps, RadioChangeEvent }
import { Button, Drawer, Radio, Space } from 'antd';

const App: React.FC = () => {
  const [open, setOpen] = useState(false);
  const [placement, setPlacement] = useState<DrawerProps['placement']>(
    'left'
  );

  const showDrawer = () => {
    setOpen(true);
  };

  const onClose = () => {
    setOpen(false);
  };

  const onChange = (e: RadioChangeEvent) => {
    setPlacement(e.target.value);
  };

  return (
    <>
      <Space>
        <Radio.Group value={placement} onChange={onChange}>
          <Radio value="top">top</Radio>
          <Radio value="right">right</Radio>
          <Radio value="bottom">bottom</Radio>
          <Radio value="left">left</Radio>
        </Radio.Group>
        <Button type="primary" onClick={showDrawer}>
          Open
        </Button>
      </Space>
      <Drawer
        title="Basic Drawer"
        placement={placement}
        closable={false}
        onClose={onClose}
        open={open}
        key={placement}
      >
        <p>Some contents...</p>
        <p>Some contents...</p>
        <p>Some contents...</p>
      </Drawer>
    </>
  );
};

export default App;
```

☐ top ☒ right ☐ bottom
☐ left

Open

Extra Actions [✎](#)

Extra actions should be placed at corner of drawer
in Ant Design, you can use `extra` prop for that.

```
import React, { useState } from 'react';
import { Button, Drawer, Radio, Space } from 'antd';
import type { DrawerProps } from 'antd/es/drawer';
import type { RadioChangeEvent } from 'antd/es/radio';

const App: React.FC = () => {
  const [open, setOpen] = useState(false);
  const [placement, setPlacement] = useState<DrawerProps['placement']>('right');

  const showDrawer = () => {
    setOpen(true);
  };

  const onChange = (e: RadioChangeEvent) => {
    setPlacement(e.target.value);
  };

  const onClose = () => {
    setOpen(false);
  };

  return (
    <div>
      <Space>
        <Radio.Group value={placement} onChange={onChange}>
          <Radio value="top">top</Radio>
          <Radio value="right">right</Radio>
          <Radio value="bottom">bottom</Radio>
          <Radio value="left">left</Radio>
        </Radio.Group>
        <Button type="primary" onClick={showDrawer}>
          Open
        </Button>
      </Space>
      <Drawer
        title="Drawer with extra actions"
        placement={placement}
        width={500}
        onClose={onClose}
        open={open}
        extra={
          <Space>
            <Button onClick={onClose}>Cancel</Button>
            <Button type="primary" onClick={onClose}>OK</Button>
          </Space>
        }
      >
        <p>Some contents...</p>
        <p>Some contents...</p>
        <p>Some contents...</p>
      </Drawer>
    </div>
  );
};

export default App;
```

Render in this

Open

Render in current dom [✎](#)

Render in current dom. custom container, check

`getContainer` .

Note: `style` and `className` props are moved to Drawer panel in v5 which is aligned with Modal component. Original `style` and `className` props are replaced by `rootStyle` and `rootClassName` .

```
import React, { useState } from 'react';
import { Button, Drawer, theme } from 'antd';

const App: React.FC = () => {
  const { token } = theme.useToken();
  const [open, setOpen] = useState(false);

  const showDrawer = () => {
    setOpen(true);
  };

  const onClose = () => {
    setOpen(false);
  };

  const containerStyle: React.CSSProperties = {
    position: 'relative',
    height: 200,
    padding: 48,
    overflow: 'hidden',
    textAlign: 'center',
    background: token.colorFillAlter,
    border: `1px solid ${token.colorBorderSecondary}`,
    borderRadius: token.borderRadiusLG,
  };

  return (
    <div style={containerStyle}>
      Render in this
      <div style={{ marginTop: 16 }}>
        <Button type="primary" onClick={showDrawer}>
          Open
        </Button>
      </div>
      <Drawer
        title="Basic Drawer"
        placement="right"
        closable={false}
        onClose={onClose}
        open={open}
        getContainer={false}
      >
        <p>Some contents...</p>
      </Drawer>
    </div>
  );
};
```

New account

Submit form in drawer

Use a form in Drawer with a submit button.

```
import React, { useState } from 'react';
import { PlusOutlined } from '@ant-design/ico
import { Button, Col, DatePicker, Drawer, Form

const { Option } = Select;

const App: React.FC = () => {
  const [open, setOpen] = useState(false);

  const showDrawer = () => {
    setOpen(true);
  };

  const onClose = () => {
    setOpen(false);
  };

  return (
    <
      <Button type="primary" onClick={showDraw
        New account
      </Button>
      <Drawer
        title="Create a new account"
        width={720}
        onClose={onClose}
        open={open}
        bodyStyle={{ paddingBottom: 80 }}
        extra={
          <Space>
            <Button onClick={onClose}>Cancel</
            <Button onClick={onClose} type="pr
              Submit
            </Button>
          </Space>
        </Drawer>
      </
    >
    <Form layout="vertical" hideRequiredMc
      <Row gutter={16}>
        <Col span={12}>
          <Form.Item
            name="name"
            label="Name"
            rules={[{ required: true, mess
          >
            <Input placeholder="Please ent
          </Form.Item>
        </Col>
        <Col span={12}>
          <Form.Item
            name="url"
            label="Url"
            rules={[{ required: true, mess
          >
            <Input
              style={{ width: '100%' }}
              addonBefore="http://"
              addonAfter=".com"
              placeholder="Please enter ur
            </>
          </Form.Item>
        </Col>
      </Row>
    <Row gutter={16}>
```



Lily

Progresser

XTech

[View Profile](#)



Lily

Progresser

XTech

[View Profile](#)

Preview drawer

Use Drawer to quickly preview details of an object, such as those in a list.

```
import React, { useState } from 'react';
import { Avatar, Col, Divider, Drawer, List, F

interface DescriptionItemProps {
  title: string;
  content: React.ReactNode;
}

const DescriptionItem = ({ title, content }: I
  <div className="site-description-item-profil
    <p className="site-description-item-profil
      {content}
    </p>
  </div>
);

const App: React.FC = () => {
  const [open, setOpen] = useState(false);

  const showDrawer = () => {
    setOpen(true);
  };

  const onClose = () => {
    setOpen(false);
  };

  return (
    <
      <List
        dataSource={[
          {
            id: 1,
            name: 'Lily',
          },
          {
            id: 2,
            name: 'Lily',
          },
        ]}
        bordered
        renderItem={(item) => (
          <List.Item
            key={item.id}
            actions={[
              <a onClick={showDrawer} key={`a-
                View Profile
              </a>,
            ]}
          >
            <List.Item.Meta
              avatar={
                <Avatar src="https://gw.alipay
              }
              title={<a href="https://ant.desi
                description="Progresser XTech"
```

Open drawer

Multi-level drawer [↗](#)

Open a new drawer on top of an existing drawer to handle multi branch tasks.

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

const App: React.FC = () => {
  const [open, setOpen] = useState(false);
  const [childrenDrawer, setChildrenDrawer] =
    useState(false);

  const showDrawer = () => {
    setOpen(true);
  };

  const onClose = () => {
    setOpen(false);
  };

  const showChildrenDrawer = () => {
    setChildrenDrawer(true);
  };

  const onChildrenDrawerClose = () => {
    setChildrenDrawer(false);
  };

  return (
    <div>
      <Button type="primary" onClick={showDrawer}>
        Open drawer
      </Button>
      <Drawer title="Multi-level drawer" width=320>
        <Button type="primary" onClick={showChildrenDrawer}>
          Two-level drawer
        </Button>
        <Drawer
          title="Two-level Drawer"
          width=320
          closable=false
          onClose={onChildrenDrawerClose}
          open={childrenDrawer}
        >
          This is two-level drawer
        </Drawer>
      </Drawer>
    </div>
  );
};

export default App;
```

Open Default Size (378px)

Open Large Size (736px)

Preset size [↗](#)

The default width (or height) of Drawer is 378px ,
and there is a preset large size 736px .

```
import React, { useState } from 'react';
import { Button, Drawer, Space } from 'antd';
import type { DrawerProps } from 'antd/es/drawer';

const App: React.FC = () => {
  const [open, setOpen] = useState(false);
  const [size, setSize] = useState<DrawerProps>({
    size: 'default'
  });

  const showDefaultDrawer = () => {
    setSize('default');
    setOpen(true);
  };

  const showLargeDrawer = () => {
    setSize('large');
    setOpen(true);
  };

  const onClose = () => {
    setOpen(false);
  };

  return (
    <div>
      <Space>
        <Button type="primary" onClick={showDefaultDrawer}>
          Open Default Size (378px)
        </Button>
        <Button type="primary" onClick={showLargeDrawer}>
          Open Large Size (736px)
        </Button>
      </Space>
      <Drawer
        title={`${size} Drawer`}
        placement="right"
        size={size}
        onClose={onClose}
        open={open}
        extra={
          <Space>
            <Button onClick={onClose}>Cancel</Button>
            <Button type="primary" onClick={onClose}>OK
          </Space>
        }
      >
        <p>Some contents...</p>
        <p>Some contents...</p>
        <p>Some contents...</p>
      </Drawer>
    </div>
  );
};

export default App;
```


```

        <Col span={24}>
          <DescriptionItem title="Phone Numk
        </Col>
      </Row>
    <Row>
      <Col span={24}>
        <DescriptionItem
          title="Github"
          content={
            <a href="http://github.com/ant
              github.com/ant-design/ant-de
            </a>
          }
        />
      </Col>
    </Row>
  </Drawer>
</>
);
};

export default App;

```

API

 **Note:** v5 use `rootClassName` & `rootStyle` to config wrapper style instead of `className` & `style` in v4 to align the API with Modal.

Props	Description	Type	Default	Version
<code>autoFocus</code>	Whether Drawer should get focused after open	<code>boolean</code>	<code>true</code>	4.1.0
<code>afterOpenChange</code>	Callback after the animation ends when switching drawers	<code>function(open)</code>	–	
<code>bodyStyle</code>	Style of the drawer content part	<code>CSSProperties</code>	–	
<code>className</code>	Config Drawer Panel <code>className</code> . Use <code>rootClassName</code> if want to config top dom style	<code>string</code>	–	
<code>closable</code>	Whether a close (x) button is visible on top left of the Drawer dialog or not	<code>boolean</code>	<code>true</code>	
<code>closeIcon</code>	Custom close icon	<code>ReactNode</code>	<code><CloseOutlined /></code>	
<code>contentWrapperStyle</code>	Style of the drawer wrapper of content part	<code>CSSProperties</code>	–	
<code>destroyOnClose</code>	Whether to unmount child components on closing drawer or not	<code>boolean</code>	<code>false</code>	
<code>extra</code>	Extra actions area at corner	<code>ReactNode</code>	–	4.1.0
<code>footer</code>	The footer for Drawer	<code>ReactNode</code>	–	
<code>footerStyle</code>	Style of the drawer	<code>CSSProperties</code>	–	

Props	Description	Type	Default	Version
	footer part			
forceRender	Pre-render Drawer component forcibly	boolean	false	
getContainer	mounted node and display window for Drawer	HTMLElement () => HTMLElement Selectors false	body	
headerStyle	Style of the drawer header part	CSSProperties	-	
height	Placement is <code>top</code> or <code>bottom</code> , height of the Drawer dialog	string number	378	
keyboard	Whether support press esc to close	boolean	true	
mask	Whether to show mask or not	boolean	true	
maskClosable	Clicking on the mask (area outside the Drawer) to close the Drawer or not	boolean	true	
maskStyle	Style for Drawer's mask element	CSSProperties	{}	
placement	The placement of the Drawer	<code>top</code> <code>right</code> <code>bottom</code> <code>left</code>	<code>right</code>	
push	Nested drawers push behavior	boolean { distance: string number }	{ distance: 180 }	4.1
rootClassName	The class name of the container of the Drawer dialog	string	-	
rootStyle	Style of wrapper element which contains mask compare to <code>style</code>	CSSProperties	-	
style	Style of Drawer panel. Use <code>bodyStyle</code> if want to config body only	CSSProperties	-	
size	preset size of drawer, default <code>378px</code> and large <code>736px</code>	'default' 'large'	'default'	4.1
title	The title for Drawer	ReactNode	-	

Props	Description	Type	Default	Version
open	Whether the Drawer dialog is visible or not	boolean	false	
width	Width of the Drawer dialog	string number	378	
zIndex	The <code>z-index</code> of the Drawer	number	1000	
onClose	Specify a callback that will be called when a user clicks mask, close button or Cancel button	function(e)	-	

Design Token

▼ Global Token

Token Name	Description	Type	Default Value
colorBgElevated	Container background color of the popup layer, in dark mode the color value of this token will be a little brighter than `colorBgContainer`. E.g: modal, pop-up, menu, etc.	string	<code>#ffffff</code>
colorBgMask	The background color of the mask, used to cover the content below the mask, Modal, Drawer and other components use this token	string	<code>rgba(0, 0, 0, 0.45)</code>
colorIcon	Weak action. Such as `allowClear` or Alert close button	string	<code>rgba(0, 0, 0, 0.45)</code>
colorIconHover	Weak action hover color. Such as `allowClear` or Alert close button	string	<code>rgba(0, 0, 0, 0.88)</code>
colorSplit	Used as the color of separator, this color is the same as colorBorderSecondary but with transparency.	string	<code>rgba(5, 5, 5, 0.06)</code>
colorText	Default text color which comply with W3C standards, and this color is also the darkest neutral color.	string	<code>rgba(0, 0, 0, 0.88)</code>
fontSizeLG	Large font size	number	16
fontWeightStrong	Control the font weight of heading components (such as h1, h2, h3) or selected item.	number	600
lineHeightLG	Line height of large text.	number	1.5
lineType	Border style of base components	string	solid
lineWidth	Border width of base components	number	1
marginSM	Control the margin of an element, with a medium-small size.	number	12
motionDurationMid	Motion speed, medium speed. Used for medium element animation interaction.	string	0.2s

Token Name	Description	Type	Default Value
motionDurationSlow	Motion speed, slow speed. Used for large element animation interaction.	<div>string</div>	0.3s
padding	Control the padding of the element.	<div>number</div>	16
paddingLG	Control the large padding of the element.	<div>number</div>	24
paddingXS	Control the extra small padding of the element.	<div>number</div>	8
zIndexPopupBase	Base zIndex of component like FloatButton, Affix which can be cover by large popup	<div>number</div>	1000