

Calendar

Container for displaying data in calendar form.

When To Use

When data is in the form of dates, such as schedules, timetables, prices calendar, lunar calendar. This component also supports Year/Month switch.

Examples

<div> <div>2023 ▾</div> <div>May ▾</div> <div>Month</div> <div>Year</div> </div>						
Su	Mo	Tu	We	Th	Fr	Sa
30	01	02	03	04	05	06
07	08	09	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	01	02	03
04	05	06	07	08	09	10

Basic [✎](#)

A basic calendar component with Year/Month switch.

```
import React from 'react';
import { Calendar } from 'antd';
import type { Dayjs } from 'dayjs';
import type { CalendarMode } from 'antd/es/calendar/generateCalendar';

const App: React.FC = () => {
  const onPanelChange = (value: Dayjs, mode: CalendarMode) => {
    console.log(value.format('YYYY-MM-DD'), mode);
  };

  return <Calendar onPanelChange={onPanelChange} />;
};

export default App;
```

2023

May

Month

Year

Su	Mo	Tu	We	Th	Fr	Sa
30	01	02	03	04	05	06
07	08 This is w... This is u...	09	10 This is w... This is u... This is e...	11	12	13
14	15 This is w... This is v... This is e...	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	01	02	03
04	05	06	07	08 This is w... This is u...	09	10 This is w... This is u... This is e...

Notice Calendar

This component can be rendered by using `dateCellRender` and `monthCellRender` with the data you need.

```
import React from 'react';
import type { BadgeProps } from 'antd';
import { Badge, Calendar } from 'antd';
import type { Dayjs } from 'dayjs';
import type { CellRenderInfo } from 'rc-picker/lib/interface';

const getListData = (value: Dayjs) => {
  let listData;
  switch (value.date()) {
    case 8:
      listData = [
        { type: 'warning', content: 'This is warning event.' },
        { type: 'success', content: 'This is usual event.' },
      ];
      break;
    case 10:
      listData = [
        { type: 'warning', content: 'This is warning event.' },
        { type: 'success', content: 'This is usual event.' },
        { type: 'error', content: 'This is error event.' },
      ];
      break;
    case 15:
      listData = [
        { type: 'warning', content: 'This is warning event' },
      ];
```

2023

May

Month

Year

Su	Mo	Tu	We	Th	Fr	Sa
30	01	02	03	04	05	06
07	08	09	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	01	02	03
04	05	06	07	08	09	10

Card

Nested inside a container element for rendering in limited space.

```
import { Calendar, theme } from 'antd';
import type { CalendarMode } from 'antd/es/calendar/generateCalendar';
import type { Dayjs } from 'dayjs';
import React from 'react';

const onPanelChange = (value: Dayjs, mode: CalendarMode) => {
  console.log(value.format('YYYY-MM-DD'), mode);
};

const App: React.FC = () => {
  const { token } = theme.useToken();

  const wrapperStyle: React.CSSProperties = {
    width: 300,
    border: `1px solid ${token.colorBorderSecondary}`,
    borderRadius: token.borderRadiusLG,
  };

  return (
    <div style={wrapperStyle}>
      <Calendar fullscreen={false} onPanelChange={onPanelChange} />
    </div>
  );
};

export default App;
```

You selected date: 2017-01-25

2017 ▾

Jan ▾

Month

Year

Su	Mo	Tu	We	Th	Fr	Sa
01	02	03	04	05	06	07
08	09	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	01	02	03	04
05	06	07	08	09	10	11

Selectable Calendar [✎](#)

A basic calendar component with Year/Month switch.

```
import React, { useState } from 'react';
import { Alert, Calendar } from 'antd';
import type { Dayjs } from 'dayjs';
import dayjs from 'dayjs';

const App: React.FC = () => {
  const [value, setValue] = useState(() => dayjs('2017-01-25'));
  const [selectedValue, setSelectedValue] = useState(() => dayjs('2017-01-25'));

  const onSelect = (newValue: Dayjs) => {
    setValue(newValue);
    setSelectedValue(newValue);
  };

  const onPanelChange = (newValue: Dayjs) => {
    setValue(newValue);
  };

  return (
    <>
      <Alert message={`You selected date: ${selectedValue?.format('YYYY-MM-DD')}`} />
      <Calendar value={value} onSelect={onSelect} onPanelChange={onPanelChange} />
    </>
  );
};
```

Custom header

Month	Year	2023	May			
Su	Mo	Tu	We	Th	Fr	Sa
30	01	02	03	04	05	06
07	08	09	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	01	02	03
04	05	06	07	08	09	10

[Customize Header](#) [✎](#)

Customize Calendar header content.

```
import React from 'react';
import dayjs from 'dayjs';
import 'dayjs/locale/zh-cn';
import type { Dayjs } from 'dayjs';
import dayLocaleData from 'dayjs/plugin/localeData';
import { Calendar, Col, Radio, Row, Select, Typography, theme } from 'antd';
import type { CalendarMode } from 'antd/es/calendar/generateCalendar';

dayjs.extend(dayLocaleData);

const App: React.FC = () => {
  const { token } = theme.useToken();

  const onPanelChange = (value: Dayjs, mode: CalendarMode) => {
    console.log(value.format('YYYY-MM-DD'), mode);
  };

  const wrapperStyle = {
    width: 300,
    border: `1px solid ${token.colorBorderSecondary}`,
    borderRadius: token.borderRadiusLG,
  };

  return (
    <div style={wrapperStyle}>
      <Calendar
        fullscreen={false}
        headerRender={({ value, type, onChange, onTypeChange }) => {
          const start = 0;
          const end = 12;
          const monthOptions = [];

          let current = value.clone();
          const localeData = value.localeData();
          const months = [];
          for (let i = 0; i < 12; i++) {
            current = current.month(i);
            months.push(localeData.monthsShort(current));
          }

          for (let i = start; i < end; i++) {
            monthOptions.push(
              <Select.Option key={i} value={i} className="month-item">
                {months[i]}
              </Select.Option>,
            );
          }
        }}
      />
    </div>
  );
}
```

```
const year = value.year();
const month = value.month();
const options = [];
for (let i = year - 10; i < year + 10; i += 1) {
  options.push(
    <Select.Option key={i} value={i} className="year-item">
      {i}
    </Select.Option>,
  );
}
return (
  <div style={{ padding: 8 }}>
    <Typography.Title level={4}>Custom header</Typography.Title>
    <Row gutter={8}>
      <Col>
        <Radio.Group
          size="small"
          onChange={(e) => onTypeChange(e.target.value)}
          value={type}
        >
          <Radio.Button value="month">Month</Radio.Button>
          <Radio.Button value="year">Year</Radio.Button>
        </Radio.Group>
      </Col>
      <Col>
        <Select
          size="small"
          dropdownMatchSelectWidth={false}
          className="my-year-select"
          value={year}
          onChange={(newYear) => {
            const now = value.clone().year(newYear);
            onChange(now);
          }}
        />
      </Col>
    </Row>
  </div>
);
```

```
// The default locale is en-US, if you want to use other locale, just set locale in entry file gl
// import dayjs from 'dayjs';
// import 'dayjs/locale/zh-cn';
// dayjs.locale('zh-cn');
```

```
<Calendar
  dateCellRender={dateCellRender}
  monthCellRender={monthCellRender}
  onPanelChange={onPanelChange}
  onSelect={onSelect}
/>
```

Property	Description	Type	Default	Version
onPanelChange={onPanelChange}	Customize the display of the date cell, the returned content will be appended to the cell	function(date: Dayjs): ReactNode	-	
dateCellRender	Customize the display of the date cell, the returned content will override the cell	function(date: Dayjs): ReactNode	-	
export default App; dateFullCellRender	The date selected by default	dayjs	-	
disabledDate	Function that specifies the dates that cannot be selected, <code>currentDate</code> is same dayjs object as	(currentDate: Dayjs) => boolean	-	

Property	Description	Type	Default	Value
	<code>value</code> prop which you shouldn't mutate it] (https://github.com/ant-design/ant-design/issues/30987)			
<code>fullscreen</code>	Whether to display in full-screen	<code>boolean</code>	<code>true</code>	
<code>headerRender</code>	Render custom header in panel	<code>function(object: {value: Dayjs, type: string, onChange: f(), onTypeChange: f()})</code>	<code>-</code>	
<code>locale</code>	The calendar's locale	<code>object</code>	(default)	
<code>mode</code>	The display mode of the calendar	<code>month</code> <code>year</code>	<code>month</code>	
<code>monthCellRender</code>	Customize the display of the month cell, the returned content will be appended to the cell	<code>function(date: Dayjs): ReactNode</code>	<code>-</code>	
<code>monthFullCellRender</code>	Customize the display of the month cell, the returned content will override the cell	<code>function(date: Dayjs): ReactNode</code>	<code>-</code>	
<code>validRange</code>	To set valid range	<code>[dayjs, dayjs]</code>	<code>-</code>	
<code>value</code>	The current selected date	<code>dayjs</code>	<code>-</code>	
<code>onChange</code>	Callback for when date changes	<code>function(date: Dayjs)</code>	<code>-</code>	
<code>onPanelChange</code>	Callback for when panel changes	<code>function(date: Dayjs, mode: string)</code>	<code>-</code>	
<code>onSelect</code>	Callback for when a date is selected	<code>function(date: Dayjs)</code>	<code>-</code>	

Design Token

▼ Global Token

Token Name	Description	Type	Default Value
<code>colorBgContainer</code>	Container background color, e.g: default button, input box, etc. Be sure not to confuse this with <code>colorBgElevated</code> .	<code>string</code>	<code>#ffffff</code>
<code>colorBgContainerDisabled</code>	Control the background color of container in disabled state.	<code>string</code>	<code>rgba(0, 0, 0, 0.04)</code>
<code>colorIcon</code>	Weak action. Such as <code>allowClear</code> or Alert close button	<code>string</code>	<code>rgba(0, 0, 0, 0.45)</code>

Token Name	Description	Type	Default Value
colorIconHover	Weak action hover color. Such as `allowClear` or Alert close button	string	<code>rgba(0, 0, 0, 0.88)</code>
colorLink	Control the color of hyperlink.	string	<code>#1677ff</code>
colorLinkActive	Control the color of hyperlink when clicked.	string	<code>#0958d9</code>
colorLinkHover	Control the color of hyperlink when hovering.	string	<code>#69b1ff</code>
colorPrimary	Brand color is one of the most direct visual elements to reflect the characteristics and communication of the product. After you have selected the brand color, we will automatically generate a complete color palette and assign it effective design semantics.	string	<code>#1677ff</code>
colorPrimaryHover	Hover state under the main color gradient.	string	<code>#4096ff</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>
colorTextDisabled	Control the color of text in disabled state.	string	<code>rgba(0, 0, 0, 0.25)</code>
colorTextHeading	Control the font color of heading.	string	<code>rgba(0, 0, 0, 0.88)</code>
colorTextLightSolid	Control the highlight color of text with background color, such as the text in Primary Button components.	string	<code>#fff</code>
borderRadiusLG	LG size border radius, used in some large border radius components, such as Card, Modal and other components.	number	8
borderRadiusSM	SM size border radius, used in small size components, such as Button, Input, Select and other input components in small size	number	4
controlHeight	The height of the basic controls such as buttons and input boxes in Ant Design	number	32
controlHeightLG	LG component height	number	40
controlHeightSM	SM component height	number	24
controlItemBgActive	Control the background color of control component item when active.	string	<code>#e6f4ff</code>
controlItemBgHover	Control the background color of control	string	<code>rgba(0, 0, 0, 0.04)</code>

Token Name	Description	Type	Default Value
	component item when hovering.		
controlPaddingHorizontal	Control the horizontal padding of an element.	number	12
fontFamily	The font family of Ant Design prioritizes the default interface font of the system, and provides a set of alternative font libraries that are suitable for screen display to maintain the readability and readability of the font under different platforms and browsers, reflecting the friendly, stable and professional characteristics.	string	-apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, 'Helvetica Neue', Arial, 'Noto Sans', sans-serif, 'Apple Color Emoji', 'Segoe UI Emoji', 'Segoe UI Symbol', 'Noto Color Emoji'
fontSize	The most widely used font size in the design system, from which the text gradient will be derived.	number	14
fontSizeLG	Large font size	number	16
fontSizeSM	Small font size	number	12
fontWeightStrong	Control the font weight of heading components (such as h1, h2, h3) or selected item.	number	600
lineHeight	Line height of text.	number	1.5714285714285714
lineHeightLG	Line height of large text.	number	1.5
lineHeightSM	Line height of small text.	number	1.6666666666666667
lineType	Border style of base components	string	solid
lineWidth	Border width of base components	number	1
lineWidthBold	The default line width of the outline class components, such as Button, Input, Select, etc.	number	2
marginXS	Control the margin of an element, with a small size.	number	8
marginXXS	Control the margin of an element, with the smallest size.	number	4
motionDurationMid	Motion speed, medium speed. Used for medium element animation interaction.	string	0.2s
motionDurationSlow	Motion speed, slow speed. Used for large element animation interaction.	string	0.3s
padding	Control the padding of the element.	number	16
paddingSM	Control the small padding of the element.	number	12
paddingXS	Control the extra small padding of the	number	8

Token Name	Description	Type	Default Value
	element.		
paddingXXS	Control the extra extra small padding of the element.	number	4
screenXS	Control the screen width of extra small screens.	number	480

FAQ

How to use Calendar with customize date library?

See [Use custom date library](#).

How to set locale for date-related components?

See [How to set locale for date-related components](#)

Date-related components locale is not working?

See FAQ [Date-related-components-locale-is-not-working?](#)