使用

ConfigProvider 使用 React 的 context 特性，只需在应用外围包裹一次即可全局生效。

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

// ...
const Demo: React.FC = () => (
  <ConfigProvider direction="rtl">
    <App />
  </ConfigProvider>
);

export default Demo;
```

Content Security Policy

部分组件为了支持波纹效果，使用了动态样式。如果开启了 Content Security Policy (CSP)，你可以通过 `csp` 属性来进行配置：

```
<ConfigProvider csp={{ nonce: 'YourNonceCode' }}>
  <Button>My Button</Button>
</ConfigProvider>
```

代码演示

国际化

```
import React, { useState } from 'react';
import { EllipsisOutlined } from '@ant-design/icons';
import type {
  ConfigProviderProps,
  RadioChangeEvent,
  TableProps,
  TourProps,
  UploadFile,
} from 'antd';
import {
  Button,
  Calendar,
  ConfigProvider,
  DatePicker,
  Divider,
  Form,
  Image,
  Input,
  InputNumber,
  Modal,
  Pagination,
  Popconfirm,
  QRCode,
  Radio,
  Select,
  Space,
  Table,
  theme,
  TimePicker,
  Tour,
  Transfer,
  Upload,
} from 'antd';
import enUS from 'antd/locale/en_US';
import zhCN from 'antd/locale/zh_CN';
import dayjs from 'dayjs';

import 'dayjs/locale/zh-cn';

type Locale = ConfigProviderProps['locale'];

dayjs.locale('en');

const { Option } = Select;
const { RangePicker } = DatePicker;

const columns: TableProps['columns'] = [
  {
    title: 'Name',
    dataIndex: 'name',
    filters: [{ text: 'filter1', value: 'filter1' }],
  },
  {
    title: 'Age',
    dataIndex: 'age',
  },
];

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

  const [open, setOpen] = useState(false);
  const [tourOpen, setTourOpen] = useState(false);
  const tourRefs = React.useRef<HTMLElement[]>([]);

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

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

  const info = () => {
    Modal.info({
      title: 'some info',
      content: 'some info',
    });
  };

  const confirm = () => {
    Modal.confirm({
      title: 'some info',
      content: 'some info',
    });
  };

  const steps: TourProps['steps'] = [
    {
      title: 'Upload File',
      description: 'Put your files here.',
      target: () => tourRefs.current[0],
    },
    {
      title: 'Save',
      description: 'Save your changes.',
      target: () => tourRefs.current[1],
    },
    {
      title: 'Other Actions',
      description: 'Click to see other actions.',
      target: () => tourRefs.current[2],
    },
  ];

  const fileList: UploadFile[] = [
    {
      uid: '-1',
      name: 'image.png',
      status: 'done',
      url:
'https://zos.alipayobjects.com/rmsportal/jkjgkEfvpUPVyRjUImniVslZfWPnJuuZ.png
    },
    {
      uid: '-2',
      percent: 50,
      name: 'image.png',
      status: 'uploading',
      url:
'https://zos.alipayobjects.com/rmsportal/jkjgkEfvpUPVyRjUImniVslZfWPnJuuZ.png
    },
    {
      uid: '-3',
      name: 'image.png',
      status: 'error',
    },
  ];

  return (
    <Space
      direction="vertical"
      size={[0, 16]}
      style={{ width: '100%', paddingTop: 16, borderTop: `1px solid
${token.colorBorder}` }}
    >
      <Pagination defaultCurrent={1} total={50} showSizeChanger />
      <Space wrap>
        <Select showSearch style={{ width: 200 }}>
          <Option value="jack">jack</Option>
          <Option value="lucy">lucy</Option>
        </Select>
        <DatePicker />
        <TimePicker />
        <RangePicker />
      </Space>
      <Space wrap>
        <Button type="primary" onClick={showModal}>
          Show Modal
        </Button>
        <Button onClick={info}>Show info</Button>
        <Button onClick={confirm}>Show confirm</Button>
        <Popconfirm title="Question?">
          <a href="#">Click to confirm</a>
        </Popconfirm>
      </Space>
      <Transfer dataSource={[]} showSearch targetKeys={[]} />
      <div style={{ width: 320, border: `1px solid ${token.colorBorder}`,
borderRadius: 8 }}>
        <Calendar fullscreen={false} value={dayjs()} />
      </div>
      <Form name="basic" autoComplete="off" labelCol={{ sm: { span: 4 } }}
wrapperCol={{ span: 6 }}>
        <Form.Item label="Username" name="username" rules={[{ required:
true }]}>
          <Input width={200} />
        </Form.Item>
        <Form.Item
          label="Age"
          name="age"
          rules={[{ type: 'number', min: 0, max: 99 }]}
          initialValue={100}
        >
          <InputNumber width={200} />
        </Form.Item>
        <Form.Item wrapperCol={{ offset: 2, span: 6 }}>
          <Button type="primary" htmlType="submit">
            Submit
          </Button>
        </Form.Item>
      </Form>
      <Table dataSource={[]} columns={columns} />
      <Modal title="Locale Modal" open={open} onCancel={hideModal}>
        <p>Locale Modal</p>
      </Modal>
      <Space wrap size={80}>
        <QRCode
          value="https://ant.design/"
          status="expired"
          onRefresh={() => console.log('refresh')}
        />
        <Image
          width={160}

src="https://zos.alipayobjects.com/rmsportal/jkjgkEfvpUPVyRjUImniVslZfWPnJuuZ
        />
      </Space>
      <Upload listType="picture-card" fileList={fileList} />
      <Divider orientation="left">Tour</Divider>
      <Button type="primary" onClick={() => setTourOpen(true)}>
        Begin Tour
      </Button>
      <Space>
        <Button
          ref={(node) => {
            node && tourRefs.current.splice(0, 0, node);
          }}
        >
          {' '}
          Upload
        </Button>
        <Button
          ref={(node) => {
            node && tourRefs.current.splice(1, 0, node);
          }}
          type="primary"
        >
          Save
        </Button>
        <Button
          ref={(node) => {
            node && tourRefs.current.splice(2, 0, node);
          }}
          icon={<EllipsisOutlined />}
        />
      </Space>
      <Tour open={tourOpen} steps={steps} onClose={() =>
setTourOpen(false)} />
    </Space>
  );
};

const App: React.FC = () => {
  const [locale, setLocal] = useState<Locale>(enUS);

  const changeLocale = (e: RadioChangeEvent) => {
    const localeValue = e.target.value;
    setLocal(localeValue);
    if (!localeValue) {
      dayjs.locale('en');
    } else {
      dayjs.locale('zh-cn');
    }
  };

  return (
    <>
      <div style={{ marginBottom: 16 }}>
```

```
        <span style={{ marginInlineEnd: 16 }}>Change locale of components: </span>
        <Radio.Group value={locale} onChange={changeLocale}>
          <Radio.Button key="en" value={enUS}>
            English
          </Radio.Button>
          <Radio.Button key="cn" value={zhCN}>
            中文
          </Radio.Button>
        </Radio.Group>
      </div>
      <ConfigProvider locale={locale}>
        <Page />
      </ConfigProvider>
    </>
  );
};

export default App;
```

方向

```
import React, { useState } from 'react';
import {
  DownloadOutlined,
  LeftOutlined,
  MinusOutlined,
  PlusOutlined,
  RightOutlined,
  SearchOutlined as SearchIcon,
  SmileOutlined,
} from '@ant-design/icons';
import type { ConfigProviderProps, RadioChangeEvent } from 'antd';
import {
  Badge,
  Button,
  Cascader,
  Col,
  ConfigProvider,
  Divider,
  Input,
  InputNumber,
  Modal,
  Pagination,
  Radio,
  Rate,
  Row,
  Select,
  Space,
  Steps,
  Switch,
  Tree,
  TreeSelect,
} from 'antd';

type DirectionType = ConfigProviderProps['direction'];

const InputGroup = Input.Group;
const ButtonGroup = Button.Group;

const { Option } = Select;
const { TreeNode } = Tree;
const { Search } = Input;

const cascaderOptions = [
  {
    value: 'tehran',
    label: 'تهران',
    children: [
      {
        value: 'tehran-c',
        label: 'تهران',
        children: [
          {
            value: 'saadat-abad',
            label: 'سعادت آباد',
          },
        ],
      },
    ],
  },
  {
    value: 'ardabil',
    label: 'اردبیل',
    children: [
      {
        value: 'ardabil-c',
        label: 'اردبیل',
        children: [
          {
            value: 'primadar',
            label: 'پیرمادر',
          },
        ],
      },
    ],
  },
  {
    value: 'gilan',
    label: 'گیلان',
    children: [
      {
        value: 'rasht',
        label: 'رشت',
        children: [
          {
            value: 'district-3',
            label: 'منطقه ۳',
          },
        ],
      },
    ],
  },
];

type Placement = 'bottomLeft' | 'bottomRight' | 'topLeft' | 'topRight';

const Page: React.FC<{ placement: Placement }> = ({ placement }) => {
  const [currentStep, setCurrentStep] = useState(0);
  const [modalOpen, setModalOpen] = useState(false);
  const [badgeCount, setBadgeCount] = useState(5);
  const [showBadge, setShowBadge] = useState(true);

  const selectBefore = (
    <Select defaultValue="Http://" style={{ width: 90 }}>
      <Option value="Http://">Http://</Option>
      <Option value="Https://">Https://</Option>
    </Select>
  );

  const selectAfter = (
    <Select defaultValue=".com" style={{ width: 80 }}>
      <Option value=".com">.com</Option>
      <Option value=".jp">.jp</Option>
      <Option value=".cn">.cn</Option>
      <Option value=".org">.org</Option>
    </Select>
  );
  // ==== Cascader ====
  const cascaderFilter = (inputValue: string, path: { label: string }[]) =>
    path.some((option) =>
option.label.toLowerCase().includes(inputValue.toLowerCase()));

  const onCascaderChange = (value: any) => {
    console.log(value);
  };
  // ==== End Cascader ====

  // ==== Modal ====
  const showModal = () => {
    setModalOpen(true);
  };

  const handleOk = (e: React.MouseEvent<HTMLElement>) => {
    console.log(e);
    setModalOpen(false);
  };

  const handleCancel = (e: React.MouseEvent<HTMLElement>) => {
    console.log(e);
    setModalOpen(false);
  };

  // ==== End Modal ====
  const onStepsChange = (newCurrentStep: number) => {
    console.log('onChange:', newCurrentStep);
    setCurrentStep(newCurrentStep);
  };

  // ==== Badge ====
  const increaseBadge = () => {
    setBadgeCount(badgeCount + 1);
  };

  const declineBadge = () => {
    setBadgeCount((prev) => (prev - 1 < 0 ? 0 : prev - 1));
  };

  const onChangeBadge = (checked: boolean) => {
    setShowBadge(checked);
  };
  // ==== End Badge ====
return (
  <div className="direction-components">
    <Row>
      <Col span={24}>
        <Divider orientation="left">Cascader example</Divider>
        <Cascader
          suffixIcon={<SearchIcon />}
          options={cascaderOptions}
          onChange={onCascaderChange}
          placeholder="یک مورد انتخاب کنید"
          placement={placement}
        />
        &nbsp;&nbsp;&nbsp;&nbsp;With search:&nbsp;&nbsp;
        <Cascader
          suffixIcon={<SmileOutlined />}
          options={cascaderOptions}
          onChange={onCascaderChange}
          placeholder="Select an item"
          placement={placement}
          showSearch={{ filter: cascaderFilter }}
        />
      </Col>
    </Row>
    <br />
    <Row>
      <Col span={12}>
        <Divider orientation="left">Switch example</Divider>
        &nbsp;&nbsp;
        <Switch defaultChecked />
        &nbsp;&nbsp;
        <Switch loading defaultChecked />
        &nbsp;&nbsp;
        <Switch size="small" loading />
      </Col>
      <Col span={12}>
        <Divider orientation="left">Radio Group example</Divider>
        <Radio.Group defaultValue="c" buttonStyle="solid">
          <Radio.Button value="a">تهران</Radio.Button>
          <Radio.Button value="b" disabled>
            اصفهان
          </Radio.Button>
          <Radio.Button value="c">فارس</Radio.Button>
          <Radio.Button value="d">خوزستان</Radio.Button>
        </Radio.Group>
      </Col>
    </Row>
```

```
      <br />
      <Row>
        <Col span={12}>
          <Divider orientation="left">Button example</Divider>
          <div className="button-demo">
            <Button type="primary" icon={<DownloadOutlined />} />
            <Button type="primary" shape="circle" icon={<DownloadOutlined />} />
            <Button type="primary" shape="round" icon={<DownloadOutlined />} />
            <Button type="primary" shape="round" icon={<DownloadOutlined />}>
              Download
            </Button>
            <Button type="primary" icon={<DownloadOutlined />}>
              Download
            </Button>
            <br />
            <Button.Group>
              <Button type="primary">
                <LeftOutlined />
                Backward
              </Button>
              <Button type="primary">
                Forward
                <RightOutlined />
              </Button>
            </Button.Group>
            <Button type="primary" loading>
              Loading
            </Button>
            <Button type="primary" size="small" loading>
              Loading
            </Button>
          </div>
        </Col>
        <Col span={12}>
          <Divider orientation="left">Tree example</Divider>
          <Tree
            showLine
            checkable
            defaultExpandedKeys={['0-0-0', '0-0-1']}
            defaultSelectedKeys={['0-0-0', '0-0-1']}
            defaultCheckedKeys={['0-0-0', '0-0-1']}
          >
            <TreeNode title="parent 1" key="0-0">
```

```
              <TreeNode title="parent 1-0" key="0-0-0" disabled>
                <TreeNode title="leaf" key="0-0-0-0" disableCheckbox />
                <TreeNode title="leaf" key="0-0-0-1" />
              </TreeNode>
              <TreeNode title="parent 1-1" key="0-0-1">
                <TreeNode title={<span style={{ color: '#1677ff'
}}>sss</span>} key="0-0-1-0" />
              </TreeNode>
            </TreeNode>
          </Tree>
        </Col>
      </Row>
      <br />
      <Row>
        <Col span={24}>
          <Divider orientation="left">Input (Input Group) example</Divider>
          <InputGroup size="large">
            <Row gutter={8}>
              <Col span={5}>
                <Input defaultValue="0571" />
              </Col>
              <Col span={8}>
                <Input defaultValue="26888888" />
              </Col>
            </Row>
          </InputGroup>
          <br />
          <InputGroup compact>
            <Input style={{ width: '20%' }} defaultValue="0571" />
            <Input style={{ width: '30%' }} defaultValue="26888888" />
          </InputGroup>
          <br />
          <InputGroup compact>
            <Select defaultValue="Option1">
              <Option value="Option1">Option1</Option>
              <Option value="Option2">Option2</Option>
            </Select>
            <Input style={{ width: '50%' }} defaultValue="input content" />
            <InputNumber />
          </InputGroup>
          <br />
          <Search placeholder="input search text" enterButton="Search"
size="large" />
          <br />
          <br />
          <div style={{ marginBottom: 16 }}>
```

```
            <Input addonBefore={selectBefore} addonAfter={selectAfter}
defaultValue="mysite" />
          </div>
          <br />
          <Row>
            <Col span={12}>
              <Divider orientation="left">Select example</Divider>
              <Space wrap>
                <Select mode="multiple" defaultValue="مورچه" style={{
width: 120 }}>
                  <Option value="jack">Jack</Option>
                  <Option value="مورچه">مورچه</Option>
                  <Option value="disabled" disabled>
                    Disabled
                  </Option>
                  <Option value="Yiminghe">yiminghe</Option>
                </Select>
                <Select defaultValue="مورچه" style={{ width: 120 }}
disabled>
                  <Option value="مورچه">مورچه</Option>
                </Select>
                <Select defaultValue="مورچه" style={{ width: 120 }}
loading>
                  <Option value="مورچه">مورچه</Option>
                </Select>
                <Select showSearch style={{ width: 200 }}
placeholder="Select a person">
                  <Option value="jack">Jack</Option>
                  <Option value="سعید">سعید</Option>
                  <Option value="tom">Tom</Option>
                </Select>
              </Space>
            </Col>
            <Col span={12}>
              <Divider orientation="left">TreeSelect example</Divider>
              <TreeSelect
                showSearch
                style={{ width: '100%' }}
                dropdownStyle={{ maxHeight: 400, overflow: 'auto' }}
                placeholder="Please select"
                allowClear
                treeDefaultExpandAll
              >
                <TreeNode title="parent 1" key="0-1">
                  <TreeNode title="parent 1-0" key="0-1-1">
                    <TreeNode title="my leaf" key="random" />
                    <TreeNode title="your leaf" key="random1" />
                  </TreeNode>
                  <TreeNode title="parent 1-1" key="random2">
                    <TreeNode title={<b style={{ color: '#08c' }}>sss</b>}
key="random3" />
                  </TreeNode>
                </TreeNode>
              </TreeSelect>
            </Col>
          </Row>
          <br />
          <Row>
            <Col span={24}>
              <Divider orientation="left">Modal example</Divider>
              <Button type="primary" onClick={showModal}>
                Open Modal
              </Button>
              <Modal title="پنجره ساده" open={modalOpen} onOk={handleOk}
onCancel={handleCancel}>
                <p>نگاشته های خود را اینجا قراردهید</p>
                <p>نگاشته های خود را اینجا قراردهید</p>
                <p>نگاشته های خود را اینجا قراردهید</p>
              </Modal>
            </Col>
          </Row>
          <br />
          <Row>
            <Col span={24}>
              <Divider orientation="left">Steps example</Divider>
              <Steps
                progressDot
                current={currentStep}
                items={[
                  {
                    title: 'Finished',
                    description: 'This is a description.',
                  },
                  {
                    title: 'In Progress',
                    description: 'This is a description.',
                  },
                  {
                    title: 'Waiting',
                    description: 'This is a description.',
                  },
                ]}
              />
              <br />
              <Steps
                current={currentStep}
                onChange={onStepsChange}
                items={[
                  {
                    title: 'Step 1',
                    description: 'This is a description.',
                  },
                  {
                    title: 'Step 2',
                    description: 'This is a description.',
                  },
                  {
                    title: 'Step 3',
                    description: 'This is a description.',
                  },
                ]}
              />
            </Col>
          </Row>
          <br />
          <Row>
            <Col span={12}>
              <Divider orientation="left">Rate example</Divider>
              <Rate defaultValue={2.5} />
              <br />
              <strong>* Note:</strong> Half star not implemented in RTL
direction, it will be
              supported after{' '}
              <a href="https://github.com/react-component/rate"
target="_blank" rel="noreferrer">
                rc-rate
              </a>{' '}
              implement rtl support.
            </Col>
            <Col span={12}>
              <Divider orientation="left">Badge example</Divider>
              <Badge count={badgeCount}>
                <a href="#" className="head-example" />
              </Badge>
              <ButtonGroup>
                <Button onClick={declineBadge}>
                  <MinusOutlined />
                </Button>
                <Button onClick={increaseBadge}>
                  <PlusOutlined />
                </Button>
              </ButtonGroup>
              <div style={{ marginTop: 12 }}>
                <Badge dot={showBadge}>
                  <a href="#" className="head-example" />
                </Badge>
                <Switch onChange={onChangeBadge} checked={showBadge} />
              </div>
            </Col>
          </Row>
        </Col>
      </Row>
      <br />
      <br />
      <Row>
        <Col span={24}>
          <Divider orientation="left">Pagination example</Divider>
          <Pagination showSizeChanger defaultCurrent={3} total={500} />
        </Col>
      </Row>
      <br />
      <Row>
        <Col span={24}>
          <Divider orientation="left">Grid System example</Divider>
          <div className="grid-demo">
            <div className="code-box-demo">
              <p>
                <strong>* Note:</strong> Every calculation in RTL grid
system is from right side
                (offset, push, etc.)
              </p>
              <Row>
                <Col span={8}>col-8</Col>
                <Col span={8} offset={8}>
                  col-8
                </Col>
              </Row>
              <Row>
                <Col span={6} offset={6}>
                  col-6 col-offset-6
                </Col>
                <Col span={6} offset={6}>
                  col-6 col-offset-6
                </Col>
              </Row>
              <Row>
                <Col span={12} offset={6}>
                  col-12 col-offset-6
                </Col>
              </Row>
              <Row>
                <Col span={18} push={6}>
                  col-18 col-push-6
                </Col>
                <Col span={6} pull={18}>
                  col-6 col-pull-18
                </Col>
              </Row>
            </div>
          </div>
        </Col>
      </Row>
    </div>
  );
};

const App: React.FC = () => {
  const [direction, setDirection] = useState<DirectionType>('ltr');
  const [placement, setPlacement] = useState<Placement>('bottomLeft');

  const changeDirection = (e: RadioChangeEvent) => {
    const directionValue = e.target.value;
    setDirection(directionValue);
    setPlacement(directionValue === 'rtl' ? 'bottomRight' : 'bottomLeft');
  };

  return (
    <>
      <div style={{ marginBottom: 16 }}>
        <span style={{ marginInlineEnd: 16 }}>Change direction of
components:</span>
        <Radio.Group defaultValue="ltr" onChange={changeDirection}>
          <Radio.Button key="ltr" value="ltr">
            LTR
          </Radio.Button>
          <Radio.Button key="rtl" value="rtl">
            RTL
          </Radio.Button>
        </Radio.Group>
      </div>
      <ConfigProvider direction={direction}>
        <Page placement={placement} />
      </ConfigProvider>
    </>
  );
};

export default App;
```

组件尺寸

```
import React, { useState } from 'react';
import {
  Button,
  Card,
  ConfigProvider,
  DatePicker,
  Divider,
  Input,
  Radio,
  Select,
  Space,
  Table,
  Tabs,
} from 'antd';
import type { ConfigProviderProps } from 'antd';

type SizeType = ConfigProviderProps['componentSize'];

const App: React.FC = () => {
  const [componentSize, setComponentSize] = useState<SizeType>('small');

  return (
    <>
      <Radio.Group
        value={componentSize}
        onChange={(e) => {
          setComponentSize(e.target.value);
        }}
      >
        <Radio.Button value="small">Small</Radio.Button>
        <Radio.Button value="middle">Middle</Radio.Button>
        <Radio.Button value="large">Large</Radio.Button>
      </Radio.Group>
      <Divider />
      <ConfigProvider componentSize={componentSize}>
```

```
        <Space size={[0, 16]} style={{ width: '100%' }}
direction="vertical">
          <Input />
          <Tabs
            defaultActiveKey="1"
            items={[
              {
                label: 'Tab 1',
                key: '1',
                children: 'Content of Tab Pane 1',
              },
              {
                label: 'Tab 2',
                key: '2',
                children: 'Content of Tab Pane 2',
              },
              {
                label: 'Tab 3',
                key: '3',
                children: 'Content of Tab Pane 3',
              },
            ]}
          />
          <Input.Search allowClear />
          <Input.TextArea allowClear />
          <Select defaultValue="demo" options={[{ value: 'demo' }]} />
          <DatePicker />
          <DatePicker.RangePicker />
          <Button>Button</Button>
          <Card title="Card">
            <Table
              columns={[
                { title: 'Name', dataIndex: 'name' },
                { title: 'Age', dataIndex: 'age' },
              ]}
              dataSource={[
                { key: '1', name: 'John Brown', age: 32 },
                { key: '2', name: 'Jim Green', age: 42 },
                { key: '3', name: 'Joe Black', age: 32 },
              ]}
            />
          </Card>
        </Space>
      </ConfigProvider>
    </>
  );
```

```
};

export default App;
```

主题

```
import React from 'react';
import {
  Button,
  ColorPicker,
  ConfigProvider,
  Divider,
  Form,
  Input,
  InputNumber,
  Space,
  Switch,
} from 'antd';
import type { ColorPickerProps, GetProp } from 'antd';

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

type ThemeData = {
  borderRadius: number;
  colorPrimary: string;
  Button?: {
    colorPrimary: string;
    algorithm?: boolean;
  };
};

const defaultData: ThemeData = {
  borderRadius: 6,
  colorPrimary: '#1677ff',
  Button: {
    colorPrimary: '#00B96B',
  },
};

export default () => {
  const [form] = Form.useForm();

  const [data, setData] = React.useState<ThemeData>(defaultData);

  return (
    <div>
```

```
      <ConfigProvider
        theme={{
          token: {
            colorPrimary: data.colorPrimary,
            borderRadius: data.borderRadius,
          },
          components: {
            Button: {
              colorPrimary: data.Button?.colorPrimary,
              algorithm: data.Button?.algorithm,
            },
          },
        }}
      >
        <Space>
          <Input />
          <Button type="primary">Button</Button>
        </Space>
      </ConfigProvider>
      <Divider />
      <Form
        form={form}
        onValuesChange={(_, allValues) => {
          setData({
            ...allValues,
          });
        }}
        name="theme"
        initialValues={defaultData}
        labelCol={{ span: 4 }}
        wrapperCol={{ span: 20 }}
      >
        <Form.Item
          name="colorPrimary"
          label="Primary Color"
          trigger="onChangeComplete"
          getValueFromEvent={(color: Color) => color.toHexString()}
        >
          <ColorPicker />
        </Form.Item>
        <Form.Item name="borderRadius" label="Border Radius">
          <InputNumber />
        </Form.Item>
        <Form.Item label="Button">
          <Form.Item name={['Button', 'algorithm']} valuePropName="checked" label="algorithm">
```

```
            <Switch />
          </Form.Item>
          <Form.Item
            name={['Button', 'colorPrimary']}
            label="Primary Color"
            trigger="onChangeComplete"
            getValueFromEvent={(color: Color) => color.toHexString()}
          >
            <ColorPicker />
          </Form.Item>
        </Form.Item>
        <Form.Item name="submit" wrapperCol={{ offset: 4, span: 20 }}>
          <Button type="primary">Submit</Button>
        </Form.Item>
      </Form>
    </div>
  );
};
```

自定义波纹

```
import React from 'react';
import { HappyProvider } from '@ant-design/happy-work-theme';
import { Button, ConfigProvider, Space } from 'antd';
import type { ConfigProviderProps, GetProp } from 'antd';

type WaveConfig = GetProp<ConfigProviderProps, 'wave'>;

// Prepare effect holder
const createHolder = (node: HTMLElement) => {
  const { borderWidth } = getComputedStyle(node);
  const borderWidthNum = parseInt(borderWidth, 10);

  const div = document.createElement('div');
  div.style.position = 'absolute';
  div.style.inset = `-${borderWidthNum}px`;
  div.style.borderRadius = 'inherit';
  div.style.background = 'transparent';
  div.style.zIndex = '999';
  div.style.pointerEvents = 'none';
  div.style.overflow = 'hidden';
  node.appendChild(div);

  return div;
};
```

```
const createDot = (holder: HTMLElement, color: string, left: number, top: number, size = 0) => {
  const dot = document.createElement('div');
  dot.style.position = 'absolute';
  dot.style.left = `${left}px`;
  dot.style.top = `${top}px`;
  dot.style.width = `${size}px`;
  dot.style.height = `${size}px`;
  dot.style.borderRadius = '50%';
  dot.style.background = color;
  dot.style.transform = 'translate(-50%, -50%)';
  dot.style.transition = 'all 1s ease-out';
  holder.appendChild(dot);

  return dot;
};

// Inset Effect
const showInsetEffect: WaveConfig['showEffect'] = (node, { event, component }) => {
  if (component !== 'Button') {
    return;
  }

  const holder = createHolder(node);

  const rect = holder.getBoundingClientRect();

  const left = event.clientX - rect.left;
  const top = event.clientY - rect.top;

  const dot = createDot(holder, 'rgba(255, 255, 255, 0.65)', left, top);

  // Motion
  requestAnimationFrame(() => {
    dot.ontransitionend = () => {
      holder.remove();
    };

    dot.style.width = '200px';
    dot.style.height = '200px';
    dot.style.opacity = '0';
  });
};

// Shake Effect
```

```
const showShakeEffect: WaveConfig['showEffect'] = (node, { component }) =>
{
  if (component !== 'Button') {
    return;
  }

  const seq = [0, -15, 15, -5, 5, 0];
  const itv = 10;

  let steps = 0;

  function loop() {
    cancelAnimationFrame((node as any).effectTimeout);

    (node as any).effectTimeout = requestAnimationFrame(() => {
      const currentStep = Math.floor(steps / itv);
      const current = seq[currentStep];
      const next = seq[currentStep + 1];

      if (!next) {
        node.style.transform = '';
        node.style.transition = '';
        return;
      }

      // Trans from current to next by itv
      const angle = current + ((next - current) / itv) * (steps % itv);

      node.style.transform = `rotate(${angle}deg)`;
      node.style.transition = 'none';

      steps += 1;
      loop();
    });
  }

  loop();
};

// Component
const Wrapper = ({ name, ...wave }: WaveConfig & { name: string }) => (
  <ConfigProvider wave={wave}>
    <Button type="primary">{name}</Button>
  </ConfigProvider>
);
```

```
const App = () => (
  <Space style={{ padding: 24 }} size="large">
    <Wrapper name="Disabled" disabled />
    <Wrapper name="Default" />
    <Wrapper name="Inset" showEffect={showInsetEffect} />
    <Wrapper name="Shake" showEffect={showShakeEffect} />
    <HappyProvider>
      <Button type="primary">Happy Work</Button>
    </HappyProvider>
  </Space>
);

export default App;
```

静态方法

```
import React, { useContext, useLayoutEffect } from 'react';
import { StyleProvider } from '@ant-design/cssinjs';
import { ExclamationCircleFilled } from '@ant-design/icons';
import { App, Button, ConfigProvider, message, Modal, notification, Space }
from 'antd';

const Demo: React.FC = () => {
  const { locale, theme } = useContext(ConfigProvider.ConfigContext);
  useLayoutEffect(() => {
    ConfigProvider.config({
      holderRender: (children) => (
        <StyleProvider hashPriority="high">
          <ConfigProvider prefixCls="static" iconPrefixCls="icon" locale=
{locale} theme={theme}>
            <App message={{ maxCount: 1 }} notification={{ maxCount: 1 }}>
              {children}
            </App>
          </ConfigProvider>
        </StyleProvider>
      ),
    });
  }, [locale, theme]);

  return (
    <div>
      <Space>
        <Button
          type="primary"
          onClick={() => {
            message.info('This is a normal message');
          }}
        >
          message
        </Button>
        <Button
          type="primary"
          onClick={() => {
            notification.open({
              message: 'Notification Title',
              description:
                'This is the content of the notification. This is the
content of the notification. This is the content of the notification.',
            });
          }}
        >
          notification
        </Button>
        <Button
          type="primary"
          onClick={() => {
            Modal.confirm({
              title: 'Do you want to delete these items?',
              icon: <ExclamationCircleFilled />,
              content: 'Some descriptions',
            });
          }}
        >
          Modal
        </Button>
      </Space>
    </div>
  );
};

export default Demo;
```

前缀

Debug

```
import React, { useState } from 'react';
import { SmileOutlined } from '@ant-design/icons';
import { Button, Checkbox, ConfigProvider, Radio, Select } from 'antd';

// Ant Design site use `es` module for view
// but do not replace related lib `lib` with `es`
// which do not show correct in site.
// We may need do convert in site also.
const App: React.FC = () => {
  const [prefixCls, setPrefixCls] = useState('light');
  return (
    <>
      <Button
        style={{ marginBottom: 12 }}
        type="primary"
        onClick={() => setPrefixCls(prefixCls === 'light' ? 'dark' :
'light')}
      >
        toggle prefixCls
      </Button>
      <br />
      <ConfigProvider prefixCls={prefixCls} iconPrefixCls="bamboo">
        <SmileOutlined />
        <Select style={{ width: 120 }} />
        <Radio>test</Radio>
        <Checkbox>test</Checkbox>
      </ConfigProvider>
    </>
  );
};

export default App;
```

获取配置

Debug

```
import React, { useState } from 'react';
import { Checkbox, ConfigProvider, Divider, Form, Input, Radio, Space }
from 'antd';
import type { ConfigProviderProps } from 'antd';

type SizeType = ConfigProviderProps['componentSize'];

const ConfigDisplay = () => {
  const { componentDisabled, componentSize } = ConfigProvider.useConfig();

  return (
    <>
      <Form.Item label="componentSize value">
        <Input value={componentSize} />
      </Form.Item>
      <Form.Item label="componentDisabled value">
        <Input value={String(componentDisabled)} disabled=
{componentDisabled} />
      </Form.Item>
    </>
  );
};

const App: React.FC = () => {
  const [componentSize, setComponentSize] = useState<SizeType>('small');
  const [disabled, setDisabled] = useState<boolean>(true);

  return (
    <div>
      <Space>
        <Radio.Group
          value={componentSize}
          onChange={(e) => {
            setComponentSize(e.target.value);
          }}
        >
          <Radio.Button value="small">Small</Radio.Button>
          <Radio.Button value="middle">Middle</Radio.Button>
          <Radio.Button value="large">Large</Radio.Button>
        </Radio.Group>
        <Checkbox checked={disabled} onChange={(e) =>
setDisabled(e.target.checked)}>
          Form disabled
        </Checkbox>
      </Space>
      <Divider />
      <ConfigProvider componentSize={componentSize}>
        <div className="example">
          <Form disabled={disabled}>
            <ConfigDisplay />
          </Form>
        </div>
      </ConfigProvider>
    </div>
  );
};

export default App;
```

警告

Debug

```
import React from 'react';
import { Alert, ConfigProvider, Input, Typography } from 'antd';

const App: React.FC = () => (
  <>
    <Typography.Title level={4}>Open single page to check the console</Typography.Title>
    <ConfigProvider warning={{ strict: false }}>
      <Alert closeText="deprecated" />
      <Input.Group />
    </ConfigProvider>
  </>
);

export default App;
```

API

参数	说明	类型	默认值	版
componentDisabled	设置 antd 组件禁用状态	boolean	-	4.2
componentSize	设置 antd 组件大小	`small` | `middle` | `large`	-	
csp	设置 [Content Security Policy]() 配置	{ nonce: string }	-	
direction	设置文本展示方向。 [示例]()	`ltr` | `rtl`	`ltr`	
getPopupContainer	弹出框（Select, Tooltip, Menu 等等）渲染父节点，默认渲染到 body 上。	function(triggerNode)	() => document.body	

getTargetContainer	配置 Affix、Anchor 滚动监听容器。	() => HTMLElement	() => window	4.2
iconPrefixCls	设置图标统一样式前缀	string	`anticon`	4.1
locale	语言包配置，语言包可到 antd/locale 目录下寻找	object	-	
popupMatchSelectWidth	下拉菜单和选择器同宽。默认将设置 `min-width`，当值小于选择框宽度时会被忽略。`false` 时会关闭虚拟滚动	boolean | number	-	5.5
popupOverflow	Select 类组件弹层展示逻辑，默认为可视区域滚动，可配置成滚动区域滚动	'viewport' | 'scroll'	'viewport'	5.5
prefixCls	设置统一样式前缀	string	`ant`	
renderEmpty	自定义组件空状态。参考 空状态	function(componentName: string): ReactNode	-	
theme	设置主题，参考 定制主题	Theme	-	5.0

variant	设置全局输入组件形态变体	`outlined` | `filled` | `borderless`	-	5.19
virtual	设置 `false` 时关闭虚拟滚动	boolean	-	4.3
warning	设置警告等级，`strict` 为 `false` 时会将废弃相关信息聚合为单条信息	{ strict: boolean }	-	5.10

ConfigProvider.config()

设置 `Modal`、`Message`、`Notification` 静态方法配置，只会对非 hooks 的静态方法调用生效。

```
ConfigProvider.config({
  // 5.13.0+
  holderRender: (children) => (
    <ConfigProvider
      prefixCls="ant"
      iconPrefixCls="anticon"
      theme={{ token: { colorPrimary: 'red' } }}
    >
      {children}
    </ConfigProvider>
  ),
});
```

ConfigProvider.useConfig() `5.3.0+`

`5.2.0` 版本后可用。获取父级 `Provider` 的值。如 `DisabledContextProvider`、`SizeContextProvider`。

```
const {
  componentDisabled, // 5.3.0+
  componentSize, // 5.3.0+
} = ConfigProvider.useConfig();
```

返回值	说明	类型	默认值	版本

componentDisabled	antd 组件禁用状态	boolean	-	5.3.0
componentSize	antd 组件大小状态	`small` | `middle` | `large`	-	5.3.0

组件配置

参数	说明	类型	默认值	
alert	设置 Alert 组件的通用属性	{ className?: string, style?: React.CSSProperties, closeIcon?: React.ReactNode }	-	5. 5.
anchor	设置 Anchor 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
avatar	设置 Avatar 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
badge	设置 Badge 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?: [BadgeProps["classNames"]], styles?: [BadgeProps["styles"]] }	-	5.
breadcrumb	设置 Breadcrumb 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
button	设置 Button 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?: [ButtonProps["classNames"]], styles?: [ButtonProps["styles"]], autoInsertSpace?: boolean }	-	5. au 5.
calendar	设置 Calendar 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
card	设置 Card 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?: [CardProps["classNames"]], styles?: [CardProps["styles"]] }	-	5. s
carousel	设置 Carousel 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
cascader	设置 Cascader 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
checkbox	设置 Checkbox 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.

collapse	设置 Collapse 组件的通用属性	{ className?: string, style?: React.CSSProperties, expandIcon?: (props) => ReactNode }	-	5. 5.
colorPicker	设置 ColorPicker 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
datePicker	设置 DatePicker 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
rangePicker	设置 RangePicker 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
descriptions	设置 Descriptions 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?: DescriptionsProps["classNames"], styles?: DescriptionsProps["styles"] }	-	5. s
divider	设置 Divider 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
drawer	设置 Drawer 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?: DrawerProps["classNames"], styles?: DrawerProps["styles"], closeIcon?: ReactNode }	-	5. s c
dropdown	设置 Dropdown 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
empty	设置 Empty 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?:EmptyProps["classNames"], styles?: EmptyProps["styles"] }	-	5. s
flex	设置 Flex 组件的通用属性	{ className?: string, style?: React.CSSProperties, vertical?: boolean }	-	5.
floatButtonGroup	设置 FloatButton.Group 组件的通用属性	{ closeIcon?: React.ReactNode }	-	5.
form	设置 Form 组件的通用属性	{ className?: string, style?: React.CSSProperties, validateMessages?: ValidateMessages, `requiredMark`?: boolean | `optional`, colon?: boolean, scrollToFirstError?: boolean | Options}	-	r c s 5. s
image	设置 Image 组件的通用属性	{ className?: string, style?: React.CSSProperties, preview?: {	-	5. 5.

		closeIcon?: React.ReactNode } }		
input	设置 Input 组件的通用属性	{ autoComplete?: string, className?: string, style?: React.CSSProperties, allowClear?: boolean | { clearIcon?: ReactNode } }	-	5. 5.
textArea	设置 TextArea 组件的通用属性	{ autoComplete?: string, className?: string, style?: React.CSSProperties, allowClear?: boolean | { clearIcon?: ReactNode } }	-	5.
layout	设置 Layout 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
list	设置 List 组件的通用属性	{ className?: string, style?: React.CSSProperties, item?:{ classNames: ListItemProps["classNames"], styles: ListItemProps["styles"] } }	-	5.
menu	设置 Menu 组件的通用属性	{ className?: string, style?: React.CSSProperties, expandIcon?: ReactNode | props => ReactNode }	-	5. 5.
mentions	设置 Mentions 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
message	设置 Message 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
modal	设置 Modal 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?: ModalProps["classNames"], styles?: ModalProps["styles"], closeIcon?: React.ReactNode }	-	5. s c
notification	设置 Notification 组件的通用属性	{ className?: string, style?: React.CSSProperties, closeIcon?: React.ReactNode }	-	5. 5.
pagination	设置 Pagination 组件的通用属性	{ showSizeChanger?: boolean, className?: string, style?: React.CSSProperties }	-	5.
progress	设置 Progress 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
radio	设置 Radio 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.

rate	设置 Rate 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
result	设置 Result 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
skeleton	设置 Skeleton 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
segmented	设置 Segmented 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
select	设置 Select 组件的通用属性	{ className?: string, showSearch?: boolean, style?: React.CSSProperties }	-	5.
slider	设置 Slider 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?: [SliderProps["classNames"]](), styles?: [SliderProps["styles"]]() }	-	5. s
switch	设置 Switch 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
space	设置 Space 的通用属性，参考 [Space]()	{ size: `small` | `middle` | `large` | `number`, className?: string, style?: React.CSSProperties, classNames?: [SpaceProps["classNames"]](), styles?: [SpaceProps["styles"]]() }	-	5.
splitter	设置 Splitter 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
spin	设置 Spin 组件的通用属性	{ className?: string, style?: React.CSSProperties, indicator?: React.ReactElement }	-	5. 5.
statistic	设置 Statistic 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
steps	设置 Steps 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
table	设置 Table 组件的通用属性	{ className?: string, style?: React.CSSProperties, expandable?: { expandIcon?: props => React.ReactNode } }	-	5. 5.
tabs	设置 Tabs 组件的通用属性	{ className?: string, style?: React.CSSProperties, indicator?: { size?: GetIndicatorSize, align?: `start` | `center` |	-	5. a r

		end }, moreIcon?: ReactNode, addIcon?: ReactNode, removeIcon?: ReactNode }		
tag	设置 Tag 组件的通用属性	{ className?: string, style?: React.CSSProperties, closeIcon?: React.ReactNode }	-	5. 5.
timeline	设置 Timeline 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
timePicker	设置 TimePicker 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
tour	设置 Tour 组件的通用属性	{ closeIcon?: React.ReactNode }	-	5.
tooltip	设置 Tooltip 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?:Tooltip["classNames"], styles?: Tooltip["styles"] }	-	5.
popover	设置 Popover 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?:Popover["classNames"], styles?: Popover["styles"] }	-	5.
popconfirm	设置 Popconfirm 组件的通用属性	{ className?: string, style?: React.CSSProperties, classNames?:Popconfirm["classNames"], styles?: Popconfirm["styles"] }	-	5.
transfer	设置 Transfer 组件的通用属性	{ className?: string, style?: React.CSSProperties, selectionsIcon?: React.ReactNode }	-	5. s 5.
tree	设置 Tree 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
typography	设置 Typography 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
upload	设置 Upload 组件的通用属性	{ className?: string, style?: React.CSSProperties }	-	5.
wave	设置水波纹特效	{ disabled?: boolean, showEffect?: (node: HTMLElement, info: { className, token, component }) => void }	-	5.

FAQ

如何增加一个新的语言包?

参考《增加语言包》。

为什么时间类组件的国际化 locale 设置不生效?

参考 FAQ 为什么时间类组件的国际化 locale 设置不生效? 。

配置 `getPopupContainer` 导致 Modal 报错?

相关 issue：https://github.com/ant-design/ant-design/issues/19974

当如下全局设置 `getPopupContainer` 为触发节点的 parentNode 时，由于 Modal 的用法不存在 `triggerNode`，这样会导致 `triggerNode is undefined` 的报错，需要增加一个判断条件。

```diff
 <ConfigProvider
-  getPopupContainer={triggerNode => triggerNode.parentNode}
+  getPopupContainer={node => {
+    if (node) {
+      return node.parentNode;
+    }
+    return document.body;
+  }}
 >
   <App />
 </ConfigProvider>
```

为什么 message.info、notification.open 或 Modal.confirm 等方法内的 ReactNode 无法继承 ConfigProvider 的属性? 比如 `prefixCls` 和 `theme`。

静态方法是使用 ReactDOM.render 重新渲染一个 React 根节点上，和主应用的 React 节点是脱离的。我们建议使用 useMessage、useNotification 和 useModal 来使用相关方法。原先的静态方法在 5.0 中已被废弃。

Vite 生产模式打包后国际化 locale 设置不生效?

相关 issue：#39045

由于 Vite 生产模式下打包与开发模式不同，cjs 格式的文件会多一层，需要 `zhCN.default` 来获取。推荐 Vite 用户直接从 `antd/es/locale` 目录下引入 esm 格式的 locale 文件。

prefixCls 优先级(前者被后者覆盖)

1. `ConfigProvider.config({ prefixCls: 'prefix-1' })`
2. `ConfigProvider.config({ holderRender: (children) => <ConfigProvider prefixCls="prefix-2">{children}</ConfigProvider> })`
3. `message.config({ prefixCls: 'prefix-3' })`