zh-CN

第一个对话框。

en-US

Basic modal.

```
import {
  Modal,
  DatePicker,
  Slider,
  Tree,
  Badge,
  Collapse,
  Timeline,
  Tabs,
  Anchor,
  Table,
  Card,
  Button,
  Calendar,
  Transfer,
  Switch,
  Typography,
  Dropdown,
} from 'antd';
import moment from 'moment';
import difference from 'lodash/difference';
import { DownOutlined, ClockCircleOutlined } from '@ant-design/icons';

const { Panel } = Collapse;
const { TreeNode } = Tree;
const { TabPane } = Tabs;
const { Meta } = Card;
const { Link } = Anchor;
const { Text } = Typography;

const text = `
  A dog is a type of domesticated animal.
  Known for its loyalty and faithfulness,
  it can be found as a welcome guest in many households across the world.
`;

const mockData = [];
for (let i = 0; i < 20; i++) {
  mockData.push({
    key: i.toString(),
    title: `content${i + 1}`,
    description: `description of content${i + 1}`,
    disabled: i % 3 < 1,
```

```
  });
}

const oriTargetKeys = mockData.filter(item => +item.key % 3 > 1).map(item =>
item.key);

const data = [
  {
    key: '1',
    name: 'John Brown',
    age: 32,
    address: 'New York No. 1 Lake Park',
  },
  {
    key: '2',
    name: 'Jim Green',
    age: 42,
    address: 'London No. 1 Lake Park',
  },
  {
    key: '3',
    name: 'Joe Black',
    age: 32,
    address: 'Sidney No. 1 Lake Park',
  },
  {
    key: '4',
    name: 'Jim Red',
    age: 32,
    address: 'London No. 2 Lake Park',
  },
];

const columnsTable = [
  {
    title: 'Name',
    dataIndex: 'name',
  },
  {
    title: 'Borrow',
    dataIndex: 'borrow',
  },
  {
    title: 'Repayment',
    dataIndex: 'repayment',
  },
];

const dataTable = [
  {
    key: '1',
    name: 'John Brown',
```

```
    borrow: 10,
    repayment: 33,
  },
  {
    key: '2',
    name: 'Jim Green',
    borrow: 100,
    repayment: 0,
  },
  {
    key: '3',
    name: 'Joe Black',
    borrow: 10,
    repayment: 10,
  },
  {
    key: '4',
    name: 'Jim Red',
    borrow: 75,
    repayment: 45,
  },
];

const expandedRowRender = () => {
  const columnsExpand = [
    { title: 'Date', dataIndex: 'date', key: 'date' },
    { title: 'Name', dataIndex: 'name', key: 'name' },
    {
      title: 'Status',
      key: 'state',
      render: () => (
        <span>
          <Badge status="success" />
          Finished
        </span>
      ),
    },
    { title: 'Upgrade Status', dataIndex: 'upgradeNum', key: 'upgradeNum' },
    {
      title: 'Action',
      dataIndex: 'operation',
      key: 'operation',
      render: () => (
        <span className="table-operation">
          <a>Pause</a>
          <a>Stop</a>
          <Dropdown>
            <a>
              More <DownOutlined />
            </a>
          </Dropdown>
        </span>
```

```
      ),
    },
  ];

  const dataExpand = [];
  for (let i = 0; i < 3; ++i) {
    data.push({
      key: i,
      date: '2014-12-24 23:12:00',
      name: 'This is production name',
      upgradeNum: 'Upgraded: 56',
    });
  }
  return <Table columns={columnsExpand} dataSource={dataExpand} pagination={false}
/>;
};

const columnsNest = [
  { title: 'Name', dataIndex: 'name', key: 'name' },
  { title: 'Platform', dataIndex: 'platform', key: 'platform' },
  { title: 'Version', dataIndex: 'version', key: 'version' },
  { title: 'Upgraded', dataIndex: 'upgradeNum', key: 'upgradeNum' },
  { title: 'Creator', dataIndex: 'creator', key: 'creator' },
  { title: 'Date', dataIndex: 'createdAt', key: 'createdAt' },
  { title: 'Action', key: 'operation', render: () => <a>Publish</a> },
];

const dataNest = [];
for (let i = 0; i < 3; ++i) {
  dataNest.push({
    key: i,
    name: 'Screem',
    platform: 'iOS',
    version: '10.3.4.5654',
    upgradeNum: 500,
    creator: 'Jack',
    createdAt: '2014-12-24 23:12:00',
  });
}

const columnsFixed = [
  {
    title: 'Full Name',
    width: 100,
    dataIndex: 'name',
    key: 'name',
    fixed: 'left',
  },
  {
    title: 'Age',
    width: 100,
    dataIndex: 'age',
```

```
    key: 'age',
    fixed: 'left',
  },
  { title: 'Column 1', dataIndex: 'address', key: '1' },
  { title: 'Column 2', dataIndex: 'address', key: '2' },
  { title: 'Column 3', dataIndex: 'address', key: '3' },
  { title: 'Column 4', dataIndex: 'address', key: '4' },
  { title: 'Column 5', dataIndex: 'address', key: '5' },
  { title: 'Column 6', dataIndex: 'address', key: '6' },
  { title: 'Column 7', dataIndex: 'address', key: '7' },
  { title: 'Column 8', dataIndex: 'address', key: '8' },
  {
    title: 'Action',
    key: 'operation',
    fixed: 'right',
    width: 100,
    render: () => <a>action</a>,
  },
];

const dataFixed = [
  {
    key: '1',
    name: 'John Brown',
    age: 32,
    address: 'New York Park',
  },
  {
    key: '2',
    name: 'Jim Green',
    age: 40,
    address: 'London Park',
  },
];

const TableTransfer = ({ leftColumns, rightColumns, ...restProps }) => (
  <Transfer {...restProps} showSelectAll={false}>
    {({
      direction,
      filteredItems,
      onItemSelectAll,
      onItemSelect,
      selectedKeys: listSelectedKeys,
      disabled: listDisabled,
    }) => {
      const columns = direction === 'left' ? leftColumns : rightColumns;

      const rowSelection = {
        getCheckboxProps: item => ({ disabled: listDisabled || item.disabled }),
        onSelectAll(selected, selectedRows) {
          const treeSelectedKeys = selectedRows
            .filter(item => !item.disabled)
```

```
            .map(({ key }) => key);
          const diffKeys = selected
            ? difference(treeSelectedKeys, listSelectedKeys)
            : difference(listSelectedKeys, treeSelectedKeys);
          onItemSelectAll(diffKeys, selected);
        },
        onSelect({ key }, selected) {
          onItemSelect(key, selected);
        },
        selectedRowKeys: listSelectedKeys,
      };

      return (
        <Table
          id="components-transfer-table"
          rowSelection={rowSelection}
          columns={columns}
          dataSource={filteredItems}
          size="small"
          style={{ pointerEvents: listDisabled ? 'none' : null }}
          onRow={({ key, disabled: itemDisabled }) => ({
            onClick: () => {
              if (itemDisabled || listDisabled) return;
              onItemSelect(key, !listSelectedKeys.includes(key));
            },
          })}
        />
      );
    }}
  </Transfer>
);

class App extends React.Component {
  state = {
    visible: false,
    targetKeys: oriTargetKeys,
    selectedKeys: [],
    disabled: false,
    showSearch: false,
  };

  handleDisable = disabled => {
    this.setState({
      disabled,
    });
  };

  handleTableTransferChange = nextTargetKeys => {
    this.setState({ targetKeys: nextTargetKeys });
  };

  triggerDisable = disabled => {
```

```
  this.setState({ disabled });
};

triggerShowSearch = showSearch => {
  this.setState({ showSearch });
};

handleTransferChange = nextTargetKeys => {
  this.setState({ targetKeys: nextTargetKeys });
};

handleTransferSelectChange = (sourceSelectedKeys, targetSelectedKeys) => {
  this.setState({ selectedKeys: [...sourceSelectedKeys, ...targetSelectedKeys] });
};

showModal = () => {
  this.setState({
    visible: true,
  });
};

handleOk = e => {
  console.log(e);
  this.setState({
    visible: false,
  });
};

handleCancel = e => {
  console.log(e);
  this.setState({
    visible: false,
  });
};

render() {
  const { disabled, selectedKeys, targetKeys, showSearch } = this.state;
  const columns = [
    {
      title: 'Name',
      dataIndex: 'name',
      key: 'name',
      filters: [
        { text: 'Joe', value: 'Joe' },
        { text: 'Jim', value: 'Jim' },
      ],
      filteredValue: null,
      onFilter: (value, record) => record.name.includes(value),
      sorter: (a, b) => a.name.length - b.name.length,
      sortOrder: true,
      ellipsis: true,
    },
```

```
    {
      title: 'Age',
      dataIndex: 'age',
      key: 'age',
      sorter: false,
      sortOrder: true,
      ellipsis: true,
    },
    {
      title: 'Address',
      dataIndex: 'address',
      key: 'address',
      filters: [
        { text: 'London', value: 'London' },
        { text: 'New York', value: 'New York' },
      ],
      filteredValue: null,
      onFilter: (value, record) => record.address.includes(value),
      sorter: false,
      sortOrder: true,
      ellipsis: true,
    },
  ];
  return (
    <>
      <Button type="primary" onClick={this.showModal}>
        Open Modal
      </Button>
      <Modal
        title="Basic Modal"
        visible={this.state.visible}
        onOk={this.handleOk}
        onCancel={this.handleCancel}
      >
        <Switch
          unCheckedChildren="disabled"
          checkedChildren="disabled"
          checked={disabled}
          onChange={this.handleDisable}
          style={{ marginBottom: 16 }}
        />
        <Card title="Card Title">
          <Card.Grid>Content</Card.Grid>
          <Card.Grid hoverable={false}>Content</Card.Grid>
          <Card.Grid>Content</Card.Grid>
          <Card.Grid>Content</Card.Grid>
          <Card.Grid>Content</Card.Grid>
          <Card.Grid>Content</Card.Grid>
          <Card.Grid>Content</Card.Grid>
        </Card>
        <Collapse>
          <Panel header="This is panel header 1" key="1">
```

```
            <Collapse defaultActiveKey="1">
              <Panel header="This is panel nest panel" key="1">
                <p>{text}</p>
              </Panel>
            </Collapse>
          </Panel>
          <Panel header="This is panel header 2" key="2">
            <p>{text}</p>
          </Panel>
          <Panel header="This is panel header 3" key="3">
            <p>{text}</p>
          </Panel>
        </Collapse>
        <Transfer
          dataSource={mockData}
          titles={['Source', 'Target']}
          targetKeys={targetKeys}
          selectedKeys={selectedKeys}
          onChange={this.handleTransferChange}
          onSelectChange={this.handleTransferSelectChange}
          render={item => item.title}
          disabled={disabled}
        />
        <TableTransfer
          dataSource={mockData}
          targetKeys={targetKeys}
          disabled={disabled}
          showSearch={showSearch}
          onChange={this.handleTableTransferChange}
          filterOption={(inputValue, item) =>
            item.title.indexOf(inputValue) !== -1 || item.tag.indexOf(inputValue)
!== -1
          }
          leftColumns={[
            {
              dataIndex: 'title',
              title: 'Name',
            },
            {
              dataIndex: 'description',
              title: 'Description',
            },
          ]}
          rightColumns={[
            {
              dataIndex: 'title',
              title: 'Name',
            },
          ]}
        />
        <Switch
          unCheckedChildren="disabled"
```

```
            checkedChildren="disabled"
            checked={disabled}
            onChange={this.triggerDisable}
            style={{ marginTop: 16 }}
          />
          <Switch
            unCheckedChildren="showSearch"
            checkedChildren="showSearch"
            checked={showSearch}
            onChange={this.triggerShowSearch}
            style={{ marginTop: 16 }}
          />
          <Anchor>
            <Link href="#components-anchor-demo-basic" title="Basic demo" />
            <Link href="#components-anchor-demo-static" title="Static demo" />
            <Link
              href="#components-anchor-demo-basic"
              title="Basic demo with Target"
              target="_blank"
            />
            <Link href="#API" title="API">
              <Link href="#Anchor-Props" title="Anchor Props" />
              <Link href="#Link-Props" title="Link Props" />
            </Link>
          </Anchor>
          <Tabs type="card">
            <TabPane tab="Tab 1" key="1">
              Content of Tab Pane 1
            </TabPane>
            <TabPane tab="Tab 2" key="2">
              Content of Tab Pane 2
            </TabPane>
            <TabPane tab="Tab 3" key="3">
              Content of Tab Pane 3
            </TabPane>
          </Tabs>
          <Timeline>
            <Timeline.Item>Create a services site 2015-09-01</Timeline.Item>
            <Timeline.Item>Solve initial network problems 2015-09-01</Timeline.Item>
            <Timeline.Item dot={<ClockCircleOutlined style={{ fontSize: '16px' }}
/>} color="red">
              Technical testing 2015-09-01
            </Timeline.Item>
            <Timeline.Item>Network problems being solved 2015-09-01</Timeline.Item>
          </Timeline>
          <Calendar />
          <Tree showLine switcherIcon={<DownOutlined />} defaultExpandedKeys={['0-0-
0']}>
            <TreeNode title="parent 1" key="0-0">
              <TreeNode title="parent 1-0" key="0-0-0">
                <TreeNode title="leaf" key="0-0-0-0" />
                <TreeNode title="leaf" key="0-0-0-1" />
```

```
            <TreeNode title="leaf" key="0-0-0-2" />
          </TreeNode>
          <TreeNode title="parent 1-1" key="0-0-1">
            <TreeNode title="leaf" key="0-0-1-0" />
          </TreeNode>
          <TreeNode title="parent 1-2" key="0-0-2">
            <TreeNode title="leaf" key="0-0-2-0" />
            <TreeNode title="leaf" key="0-0-2-1" />
          </TreeNode>
        </TreeNode>
      </Tree>
      <Table columns={columns} dataSource={data} footer={() => 'Footer'} />
      <Table
        columns={columnsTable}
        dataSource={dataTable}
        pagination={false}
        id="table-demo-summary"
        bordered
        summary={pageData => {
          let totalBorrow = 0;
          let totalRepayment = 0;

          pageData.forEach(({ borrow, repayment }) => {
            totalBorrow += borrow;
            totalRepayment += repayment;
          });

          return (
            <>
              <tr>
                <th>Total</th>
                <td>
                  <Text type="danger">{totalBorrow}</Text>
                </td>
                <td>
                  <Text>{totalRepayment}</Text>
                </td>
              </tr>
              <tr>
                <th>Balance</th>
                <td colSpan={2}>
                  <Text type="danger">{totalBorrow - totalRepayment}</Text>
                </td>
              </tr>
            </>
          );
        }}
      />
      <br />
      <Table columns={columnsNest} expandable={{ expandedRowRender }}
dataSource={dataNest} />
      <Table columns={columnsFixed} dataSource={dataFixed} scroll={{ x: 1300, y:
```

```
100 }} />
          <Card
            hoverable
            style={{ width: 240 }}
            cover={
              <img alt="example"
src="https://os.alipayobjects.com/rmsportal/QBnOOoLaAfKPirc.png" />
            }
          >
            <Meta title="Europe Street beat" description="www.instagram.com" />
          </Card>
          <Slider defaultValue={30} />
          <DatePicker defaultValue={moment('2015/01/01', 'YYYY/MM/DD')}
format="YYYY/MM/DD" />
          <Badge count={5}>
            <a href="#" className="head-example" />
          </Badge>
        </Modal>
      </>
    );
  }
}

export default () => <App />;
```