zh-CN

用户填写必须的信息以注册新用户。

en-US

Fill in this form to create a new account for you.

```
import React, { useState } from 'react';
import {
  Form,
  Input,
  InputNumber,
  Cascader,
  Select,
  Row,
  Col,
  Checkbox,
  Button,
  AutoComplete,
} from 'antd';

const { Option } = Select;

const residences = [
  {
    value: 'zhejiang',
    label: 'Zhejiang',
    children: [
      {
        value: 'hangzhou',
        label: 'Hangzhou',
        children: [
          {
            value: 'xihu',
            label: 'West Lake',
          },
        ],
      },
    ],
  },
  {
    value: 'jiangsu',
    label: 'Jiangsu',
    children: [
      {
        value: 'nanjing',
        label: 'Nanjing',
        children: [
          {
            value: 'zhonghuamen',
```

```
            label: 'Zhong Hua Men',
          },
        ],
      },
    ],
  },
];

const formItemLayout = {
  labelCol: {
    xs: { span: 24 },
    sm: { span: 8 },
  },
  wrapperCol: {
    xs: { span: 24 },
    sm: { span: 16 },
  },
};
const tailFormItemLayout = {
  wrapperCol: {
    xs: {
      span: 24,
      offset: 0,
    },
    sm: {
      span: 16,
      offset: 8,
    },
  },
};

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

  const onFinish = (values: any) => {
    console.log('Received values of form: ', values);
  };

  const prefixSelector = (
    <Form.Item name="prefix" noStyle>
      <Select style={{ width: 70 }}>
        <Option value="86">+86</Option>
        <Option value="87">+87</Option>
      </Select>
    </Form.Item>
  );

  const suffixSelector = (
    <Form.Item name="suffix" noStyle>
      <Select style={{ width: 70 }}>
        <Option value="USD">$</Option>
        <Option value="CNY">¥</Option>
```

```
      </Select>
    </Form.Item>
  );

  const [autoCompleteResult, setAutoCompleteResult] = useState<string[]>([]);

  const onWebsiteChange = (value: string) => {
    if (!value) {
      setAutoCompleteResult([]);
    } else {
      setAutoCompleteResult(['.com', '.org', '.net'].map(domain =>
`${value}${domain}`));
    }
  };

  const websiteOptions = autoCompleteResult.map(website => ({
    label: website,
    value: website,
  }));

  return (
    <Form
      {...formItemLayout}
      form={form}
      name="register"
      onFinish={onFinish}
      initialValues={{
        residence: ['zhejiang', 'hangzhou', 'xihu'],
        prefix: '86',
      }}
      scrollToFirstError
    >
      <Form.Item
        name="email"
        label="E-mail"
        rules={[
          {
            type: 'email',
            message: 'The input is not valid E-mail!',
          },
          {
            required: true,
            message: 'Please input your E-mail!',
          },
        ]}
      >
        <Input />
      </Form.Item>

      <Form.Item
        name="password"
        label="Password"
```

```
        rules={[
          {
            required: true,
            message: 'Please input your password!',
          },
        ]}
        hasFeedback
      >
        <Input.Password />
      </Form.Item>

      <Form.Item
        name="confirm"
        label="Confirm Password"
        dependencies={['password']}
        hasFeedback
        rules={[
          {
            required: true,
            message: 'Please confirm your password!',
          },
          ({ getFieldValue }) => ({
            validator(_, value) {
              if (!value || getFieldValue('password') === value) {
                return Promise.resolve();
              }
              return Promise.reject(new Error('The two passwords that you entered do
not match!'));
            },
          }),
        ]}
      >
        <Input.Password />
      </Form.Item>

      <Form.Item
        name="nickname"
        label="Nickname"
        tooltip="What do you want others to call you?"
        rules={[{ required: true, message: 'Please input your nickname!',
whitespace: true }]}
      >
        <Input />
      </Form.Item>

      <Form.Item
        name="residence"
        label="Habitual Residence"
        rules={[
          { type: 'array', required: true, message: 'Please select your habitual
residence!' },
        ]}
```

```
      >
        <Cascader options={residences} />
      </Form.Item>

      <Form.Item
        name="phone"
        label="Phone Number"
        rules={[{ required: true, message: 'Please input your phone number!' }]}
      >
        <Input addonBefore={prefixSelector} style={{ width: '100%' }} />
      </Form.Item>

      <Form.Item
        name="donation"
        label="Donation"
        rules={[{ required: true, message: 'Please input donation amount!' }]}
      >
        <InputNumber addonAfter={suffixSelector} style={{ width: '100%' }} />
      </Form.Item>

      <Form.Item
        name="website"
        label="Website"
        rules={[{ required: true, message: 'Please input website!' }]}
      >
        <AutoComplete options={websiteOptions} onChange={onWebsiteChange}
placeholder="website">
          <Input />
        </AutoComplete>
      </Form.Item>

      <Form.Item
        name="intro"
        label="Intro"
        rules={[{ required: true, message: 'Please input Intro' }]}
      >
        <Input.TextArea showCount maxLength={100} />
      </Form.Item>

      <Form.Item
        name="gender"
        label="Gender"
        rules={[{ required: true, message: 'Please select gender!' }]}
      >
        <Select placeholder="select your gender">
          <Option value="male">Male</Option>
          <Option value="female">Female</Option>
          <Option value="other">Other</Option>
        </Select>
      </Form.Item>

      <Form.Item label="Captcha" extra="We must make sure that your are a human.">
```

```
        <Row gutter={8}>
          <Col span={12}>
            <Form.Item
              name="captcha"
              noStyle
              rules={[{ required: true, message: 'Please input the captcha you got!'
}]}
            >
              <Input />
            </Form.Item>
          </Col>
          <Col span={12}>
            <Button>Get captcha</Button>
          </Col>
        </Row>
      </Form.Item>

      <Form.Item
        name="agreement"
        valuePropName="checked"
        rules={[
          {
            validator: (_, value) =>
              value ? Promise.resolve() : Promise.reject(new Error('Should accept
agreement')),
          },
        ]}
        {...tailFormItemLayout}
      >
        <Checkbox>
          I have read the <a href="">agreement</a>
        </Checkbox>
      </Form.Item>
      <Form.Item {...tailFormItemLayout}>
        <Button type="primary" htmlType="submit">
          Register
        </Button>
      </Form.Item>
    </Form>
  );
};

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