水平表单-栅格布局-双向数据绑定

如果是水平表单会有个栅格布局

```
const FormItem = Form.Item;

  state = {
    isVisible: false        //初始时Modal不显示
  };

  handleOperate = type => {       //当点击创建员工按钮事触发的事件 => Modal框显示
    if (type == 'create') {
      this.setState({
        type,
        isVisible: true,
        title: '创建员工'
      });
    }
  };

  // 创建员工提交
  handleSubmit = () => {
    let type = this.state.type;
    let data = this.userForm.props.form.getFieldsValue();
    axios
      .ajax({
        url: '/user/add',
        data: {
          params: data
        }
      })
      .then(res => {
        if (res.code == 0) {
          this.userForm.props.form.resetFields();     //提交的时候顺便清空表单
          this.setState({
            isVisible: false
          });
          this.requestList();
        }
      });
  };

...
```

```
render() {
  return (
    ...

    <Button
      type="primary"
      icon="plus"
      onClick={() => this.handleOperate('create')}     //当点击按钮触发事件
      >
      创建员工
    </Button>

    ...

    <Modal
      title={this.state.title}
      visible={this.state.isVisible}       //通过visible这个属性控制modal框的隐藏显示
      onOk={this.handleSubmit}          //当点击modal框中的ok按钮时绑定的事件
      onCancel={() => {  //点击modal框中cancel按钮时绑定的事件，把isVisible置为false
        this.userForm.props.form.resetFields();    //取消的时候清空表单,重置表单，自带的
方法
        this.setState({
          isVisible: false
        });
      }}
      width={600}
      >
      <UserForm              //使用下面定义好的表单模板
        type={this.state.type}
        wrappedComponentRef={inst => (this.userForm = inst)} //绑定表单，获取表单值
        />
    </Modal>
  )
}

...

//创建form模板，在上面<Modal>中调用
class UserForm extends Component {
  render() {
    const formItemLayout = {            //栅格布局
      labelCol: { span: 5 },
      wrapperCol: { span: 19 }
    };
```

```
const { getFieldDecorator } = this.props.form;    //使用这个进行表单双向数据绑定
//getFieldDecorator("user_name", {initialValue: ...})  //第二项可以传入初始值

return (
  <Form layout="horizontal">
    <FormItem label="用户名" {...formItemLayout}>
      {
        getFieldDecorator('user_name')(
          <Input type="text" placeholder="请输入用户名" />   //输入框
        )
      }
    </FormItem>
    <FormItem label="性别" {...formItemLayout}>
      {
        getFieldDecorator('sex')(
          <RadioGroup                                  //圆形选择框
            <Radio value={1}>男</Radio>
            <Radio value={2}>女</Radio>
          </RadioGroup>
        )
      }
    </FormItem>
    <FormItem label="状态" {...formItemLayout}>
      {
        getFieldDecorator('state')(
          <Select>                                     //下拉菜单
            <Option value={1}>咸鱼干</Option>
            <Option value={2}>帅哥</Option>
            <Option value={3}>女神</Option>
            <Option value={4}>创业狗</Option>
            <Option value={5}>傻逼</Option>
          </Select>
        )
      }
    </FormItem>
    <FormItem label="生日" {...formItemLayout}>
      { getFieldDecorator('birthday')(<DatePicker />) } //日期下拉框
    </FormItem>
    <FormItem label="联系地址" {...formItemLayout}>
      {
        getFieldDecorator('address')(
          <TextArea rows={3} placeholder="请输入联系地址" />  //textarea框
        )
      }
    </FormItem>
  </Form>
```

```
    );
  }
}

UserForm = Form.create({})(UserForm);
```