09 React 表单组件

受控组件 Vs. 非受控组件

- 受控组件：组件的值同步到 state，使用 value 属性
- 非受控组件：组件的值不同步到 state，使用 defaultValue 属性

input

```tsx
// 受控组件举例
import { FC, useState, ChangeEvent } from 'react';

const ControlledComponent: FC = () => {
  const [text, setText] = useState<string>('');

  const handleChange = (event: ChangeEvent<HTMLInputElement>) => {
    const { value } = event.target;
    setText(value);
  };

  const handleClick = () => {
    console.log(text);
  };

  return (
    <>
      <h2>受控组件</h2>
      <input type="text" defaultValue={text} onChange={handleChange} />
      <button onClick={handleClick}>打印</button>
    </>
  );
};

export default ControlledComponent;
```

textarea

```tsx
import { FC, useState, ChangeEvent } from 'react';

```

```tsx
const ControlledTextArea: FC = () => {
  const [text, setText] = useState<string>('');

  const handleChange = (event: ChangeEvent<HTMLTextAreaElement>) => {
    const { value } = event.target;
    setText(value);
  };

  const getHtml = () => {
    return { __html: text.replace('\n', '<br/>') };
  };

  return (
    <>
      <h2>受控组件 textarea</h2>
      <textarea value={text} onChange={handleChange}></textarea>
      <p dangerouslySetInnerHTML={getHtml()}></p>
    </>
  );
};

export default ControlledTextArea;

```

radio

```tsx
import { FC, useState, ChangeEvent } from 'react';

const ControlledRadio: FC = () => {
  const [gender, setGender] = useState<string>('male');

  const handleChange = (event: ChangeEvent<HTMLInputElement>) => {
    const { value } = event.target;
    setGender(value);
  };

  const handleClick = () => {
    console.log(gender);
  };

  return (
    <>
      <h2>受控组件 radio</h2>
      <label htmlFor="g1">Male</label>
      <input
```

```
        type="radio"
        name="gender"
        id="g1"
        checked={gender === 'male'}
        value={'male'}
        onChange={handleChange}
      />
      <label htmlFor="g2">Femal</label>
      <input
        type="radio"
        name="gender"
        id="g2"
        checked={gender === 'female'}
        value={'female'}
        onChange={handleChange}
      />
      <button onClick={handleClick}>打印</button>
    </>
  );
};

export default ControlledRadio;
```

checkbox

```
import { FC, useState, ChangeEvent } from 'react';
import { produce } from 'immer';

const ControlledCheckbox: FC = () => {
  const [checked, setChecked] = useState(false);

  const toggleChecked = () => {
    setChecked(!checked);
  };

  const [selectedCityList, setSelectedCityList] = useState<string[]>([]);

  const toggleSelectedCityList = (event: ChangeEvent<HTMLInputElement>) => {
    const { value } = event.target;
    if (selectedCityList.includes(value)) {
      // 选中 → 未选中：移除
      const index = selectedCityList.findIndex(item => item === value);
      setSelectedCityList(
        produce(draft => {
          draft.splice(index, 1);
```

```
        })
      );
    } else {
      // 未选中 → 选中：添加
      setSelectedCityList(
        produce(draft => {
          draft.push(value);
        })
      );
    }
  };

  return (
    <>
      <h2>受控组件 checkbox</h2>
      <label htmlFor="checkbox1">选中</label>
      <input type="checkbox" id="checkbox1" checked={checked} onChange={toggleChecked} />
      {JSON.stringify(checked)}
      <br />
      <br />
      <label htmlFor="c1">深圳</label>
      <input
        type="checkbox"
        id="c1"
        value="shenzhen"
        checked={selectedCityList.includes('shenzhen')}
        onChange={toggleSelectedCityList}
      />
      <label htmlFor="c2">上海</label>
      <input
        type="checkbox"
        id="c2"
        value="shanghai"
        checked={selectedCityList.includes('shanghai')}
        onChange={toggleSelectedCityList}
      />
      <label htmlFor="c3">北京</label>
      <input
        type="checkbox"
        id="c3"
        value="beijing"
        checked={selectedCityList.includes('beijing')}
        onChange={toggleSelectedCityList}
      />
      {JSON.stringify(selectedCityList)}
```

```
      <input type="hidden" name="cities" value={JSON.stringify(selectedCityList)} />
    </>
  );
};

export default ControlledCheckbox;
```

select

```
import { FC, useState, ChangeEvent } from 'react';

const ControlledSelect: FC = () => {
  const [lang, setLang] = useState('js');

  const handleChange = (event: ChangeEvent<HTMLSelectElement>) => {
    const { value } = event.target;
    setLang(value);
  };

  return (
    <>
      <h2>受控组件 select</h2>
      <select value={lang} onChange={handleChange}>
        <option value="js">javascript</option>
        <option value="py">python</option>
        <option value="go">go</option>
      </select>
      {JSON.stringify(lang)}
    </>
  );
};

export default ControlledSelect;
```

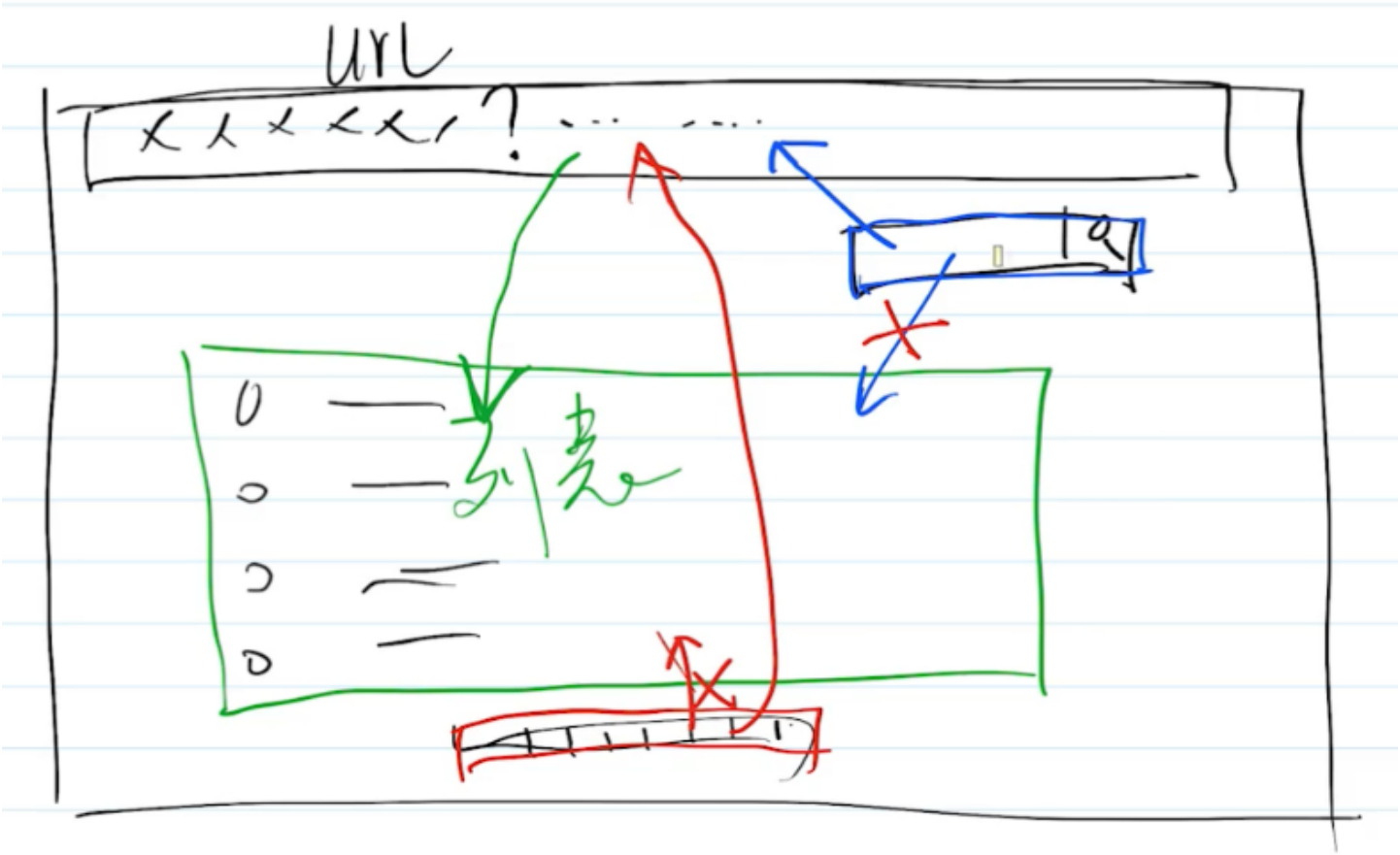
危险地设置内部 HTML

```
const markup = { __html:'<p>some raw html</p>' };
return <div dangerouslySetInnerHTML={markup} />;
```

> 这很**危险**。与底层的 DOM `innerHTML` 属性一样，你必须极度谨慎！除非标记语言来自完全可信的来源，否则通过这种方式引入 XSS 是容易被攻击的。

基于 URL 的页面交互设计 —— 解耦合

> 页面之间的组件，通过 URL 进行渲染，而不是互相通信，降低了耦合程度，使得更加易于维护。



表单验证

Antd <Form> rules

react-hook-form

Formik