

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

```
1 // 受控组件举例
2 import { FC, useState, ChangeEvent } from 'react';
3
4 const ControlledComponent: FC = () => {
5   const [text, setText] = useState<string>('');
6
7   const handleChange = (event: ChangeEvent<HTMLInputElement>) => {
8     const { value } = event.target;
9     setText(value);
10  };
11
12  const handleClick = () => {
13    console.log(text);
14  };
15
16  return (
17    <>
18      <h2>受控组件</h2>
19      <input type="text" defaultValue={text} onChange={handleChange} />
20      <button onClick={handleClick}>打印</button>
21    </>
22  );
23 };
24
25 export default ControlledComponent;
```

textarea

```
1 import { FC, useState, ChangeEvent } from 'react';
2
```

```

3  const ControlledTextArea: FC = () => {
4    const [text, setText] = useState<string>('');
5
6    const handleChange = (event: ChangeEvent<HTMLTextAreaElement>) => {
7      const { value } = event.target;
8      setText(value);
9    };
10
11    const getHtml = () => {
12      return { __html: text.replace('\n', '<br/>') };
13    };
14
15    return (
16      <>
17        <h2>受控组件 textarea</h2>
18        <textarea value={text} onChange={handleChange}></textarea>
19        <p dangerouslySetInnerHTML={getHtml()}></p>
20      </>
21    );
22  };
23
24  export default ControlledTextArea;
25

```

radio

```

1  import { FC, useState, ChangeEvent } from 'react';
2
3  const ControlledRadio: FC = () => {
4    const [gender, setGender] = useState<string>('male');
5
6    const handleChange = (event: ChangeEvent<HTMLInputElement>) => {
7      const { value } = event.target;
8      setGender(value);
9    };
10
11    const handleClick = () => {
12      console.log(gender);
13    };
14
15    return (
16      <>
17        <h2>受控组件 radio</h2>
18        <label htmlFor="g1">Male</label>
19        <input

```

```

20     type="radio"
21     name="gender"
22     id="g1"
23     checked={gender === 'male'}
24     value={'male'}
25     onChange={handleChange}
26   />
27   <label htmlFor="g2">Female</label>
28   <input
29     type="radio"
30     name="gender"
31     id="g2"
32     checked={gender === 'female'}
33     value={'female'}
34     onChange={handleChange}
35   />
36   <button onClick={handleClick}>打印</button>
37 </>
38 );
39 };
40
41 export default ControlledRadio;

```

checkbox

```

1 import { FC, useState, ChangeEvent } from 'react';
2 import { produce } from 'immer';
3
4 const ControlledCheckbox: FC = () => {
5   const [checked, setChecked] = useState(false);
6
7   const toggleChecked = () => {
8     setChecked(!checked);
9   };
10
11   const [selectedCityList, setSelectedCityList] = useState<string[]>([]);
12
13   const toggleSelectedCityList = (event: ChangeEvent<HTMLInputElement>) => {
14     const { value } = event.target;
15     if (selectedCityList.includes(value)) {
16       // 选中 → 未选中: 移除
17       const index = selectedCityList.findIndex(item => item === value);
18       setSelectedCityList(
19         produce(draft => {
20           draft.splice(index, 1);

```

```
21     })
22   );
23   } else {
24     // 未选中 → 选中: 添加
25     setSelectedCityList(
26       produce(draft => {
27         draft.push(value);
28       })
29     );
30   }
31 };
32
33 return (
34   <>
35     <h2>受控组件 checkbox</h2>
36     <label htmlFor="checkbox1">选中</label>
37     <input type="checkbox" id="checkbox1" checked={checked} onChange=
38       {toggleChecked} />
39     {JSON.stringify(checked)}
40     <br />
41     <label htmlFor="c1">深圳</label>
42     <input
43       type="checkbox"
44       id="c1"
45       value="shenzhen"
46       checked={selectedCityList.includes('shenzhen')}
47       onChange={toggleSelectedCityList}
48     />
49     <label htmlFor="c2">上海</label>
50     <input
51       type="checkbox"
52       id="c2"
53       value="shanghai"
54       checked={selectedCityList.includes('shanghai')}
55       onChange={toggleSelectedCityList}
56     />
57     <label htmlFor="c3">北京</label>
58     <input
59       type="checkbox"
60       id="c3"
61       value="beijing"
62       checked={selectedCityList.includes('beijing')}
63       onChange={toggleSelectedCityList}
64     />
65     {JSON.stringify(selectedCityList)}
```

```

66     <input type="hidden" name="cities" value=
      {JSON.stringify(selectedCityList)} />
67   </>
68   );
69 };
70
71 export default ControlledCheckbox;

```

select

```

1  import { FC, useState, ChangeEvent } from 'react';
2
3  const ControlledSelect: FC = () => {
4    const [lang, setLang] = useState('js');
5
6    const handleChange = (event: ChangeEvent<HTMLSelectElement>) => {
7      const { value } = event.target;
8      setLang(value);
9    };
10
11    return (
12      <>
13        <h2>受控组件 select</h2>
14        <select value={lang} onChange={handleChange}>
15          <option value="js">javascript</option>
16          <option value="py">python</option>
17          <option value="go">go</option>
18        </select>
19        {JSON.stringify(lang)}
20      </>
21    );
22  };
23
24  export default ControlledSelect;

```

危险地设置内部 HTML

```

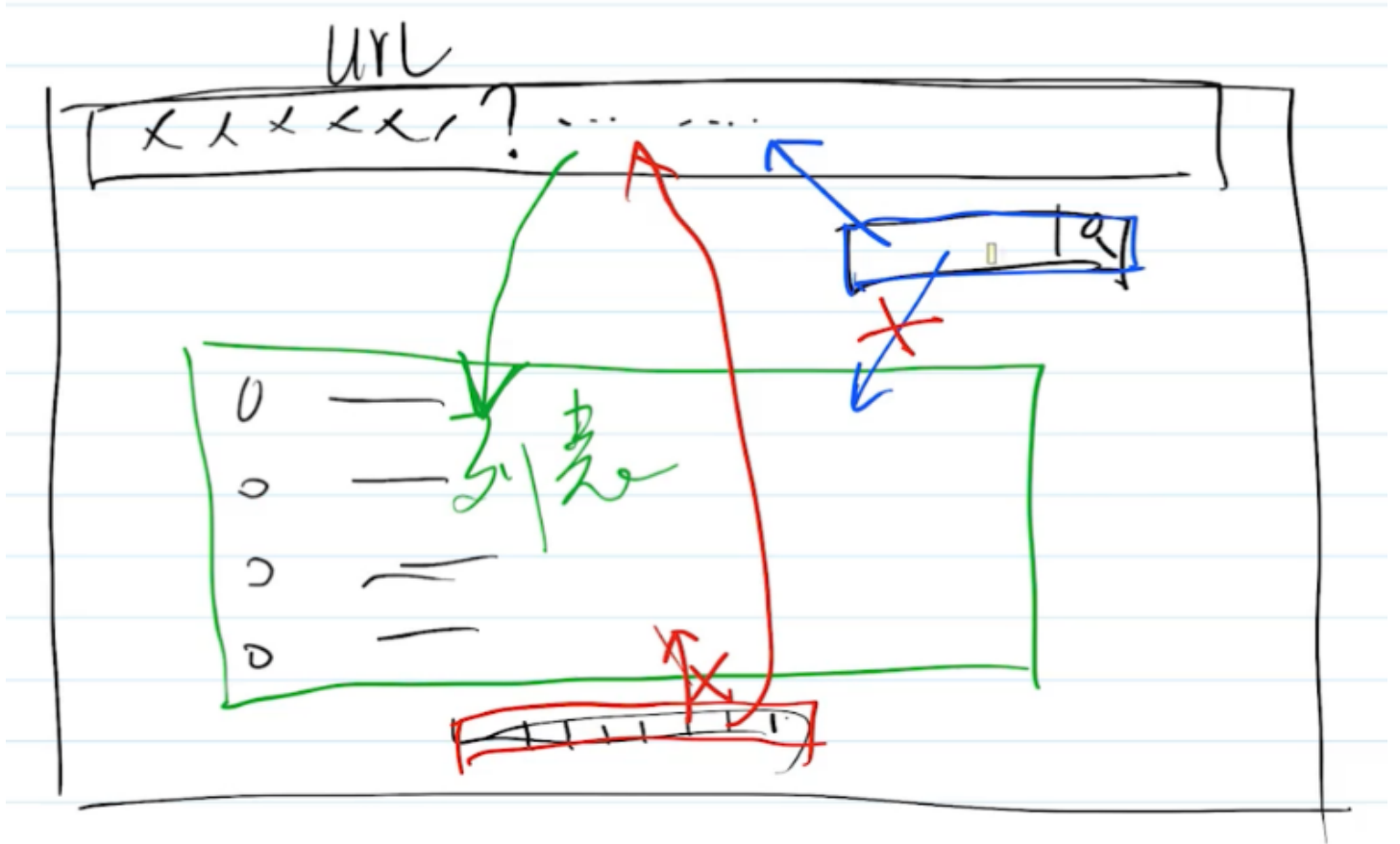
1  const markup = { __html: '<p>some raw html</p>' };
2  return <div dangerouslySetInnerHTML={markup} />;

```

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](#)