#基础知识

组件从概念上类似于 JavaScript 函数，它接受任意的入参即 **props**。

- props 是接收的组件外部传入的值
- 因为是外部传入所以验证过程是必不可少的

#实例操作

#按钮文本

下面针对上一章的用户列表组件，来给搜索设置自定义按钮文字。

App.js 设置 btnTitle 属性

```
<div className="app">
  <Add btnTitle="后盾人@向军大叔" />
  <List />
</div>

```

components/Add/index.js 使用 this.props 接收数据

```
<div className="add">
  <input className="input" />
  <button>{this.props.btnTitle}</button>
</div>

```

最终效果如图

#表格标题

下面使用 children 来设置标签的标题

App.js 中对使用的组件 List 包裹需要传递的内容

```
render() {
  return (
    <div className="app">
      <Add btnTitle="后盾人@向军大叔" />
```

```
        <List>用户列表展示</List>
      </div>
    );
}

```

components/User/index.js 使用 this.props.children 接收数据

```
render() {
  return (
    <div>
      <table border="1" width="100%">
        <caption>{this.props.children}</caption>
        <thead>
          <tr>
            <th>姓名</th>
            <th>年龄</th>
          </tr>
          <User />
        </thead>
      </table>
    </div>
  );
}

```

components/User/index.css 添加标签 caption 的样式

```
caption {
    padding: 10px;
    font-size: 1.2em;
    font-weight: bold;
}

```

最终效果如图

#类型检测

我们需要prop-types (opens new window)对传入组件的参数进行检测，检测是保证组件健状性的必要条件

#安装扩展

安装扩展包，然后重新 npm start 运行项目

```
npm i prop-types

```

为了区分函数组件与类组件的不同使用方式，现将组件 compontents/Add/index.js 改成函数组件

```
import React from "react";
import "./index.css";

export default function Add(props) {
  return (
    <div className="add">
      <input />
      <button>{props.btnTitle}</button>
    </div>
  );
}

```

#约束类型

现在要求传入搜索按钮的文本必须为字符串

```
import React from "react";
import PT from "prop-types";
import "./index.css";
export default function Add(props) {
  return (
    <div className="add">
      <input placeholder="33" />
      <button>{props.btnTitle}</button>
    </div>
  );
}
Add.propTypes = {
  btnTitle: PT.string
};

```

App.js 使用组件时传入数值类型将会在控制台产生错误

```
<Add btnTitle={99} />

```

```
▶Warning: Failed prop type: Invalid prop `btnTitle` of type `number` supplied to `Search`, expected `string`.
```

#必须传参

约束类型只会有参数时进行检测，如果 props 必须传递需要使用**isRequired**限制

```
Add.propTypes = {
  btnTitle: PT.string.isRequired
};

```

#默认值

对没有传递的参数可以使用默认值，下例中当按钮没有传递参数时使用默认值**后盾人**

```
Add.defaultProps = {
  btnTitle: "后盾人"
};

```

#类中使用

上面示例是在函数中的使用，下面来看在类中的使用方法

- 要求 props 为字符串类型
- 如果没有值时使用默认值“用户列表”

```
export default class List extends Component {
  static propTypes = {
    children: PT.string
  };
  static defaultProps = {
    children: "用户列表"
  };

```

在 App.js 中使用组件并不设置内容时，将使用默认值替代

```
export default class App extends Component {
  render() {
    return (
      <div className="app">
        <Add btnTitle="后盾人@向军大叔" />
        <List />
      </div>
    );
  }
}

```