React.js 小书

<-- 返回首页

实战分析：评论功能（四）

- 作者：胡子大哈
- 原文链接： http://huziketang.com/books/react/lesson25
- 转载请注明出处，保留原文链接和作者信息。

（本文未审核）

目前为止，第二阶段知识已经基本介绍完，我们已经具备了项目上手实战必备的 React.js 知识，现在可以把这些知识应用起来。接下来是实战环节，我们会继续上一阶段的例子，把评论功能做得更加复杂一点。

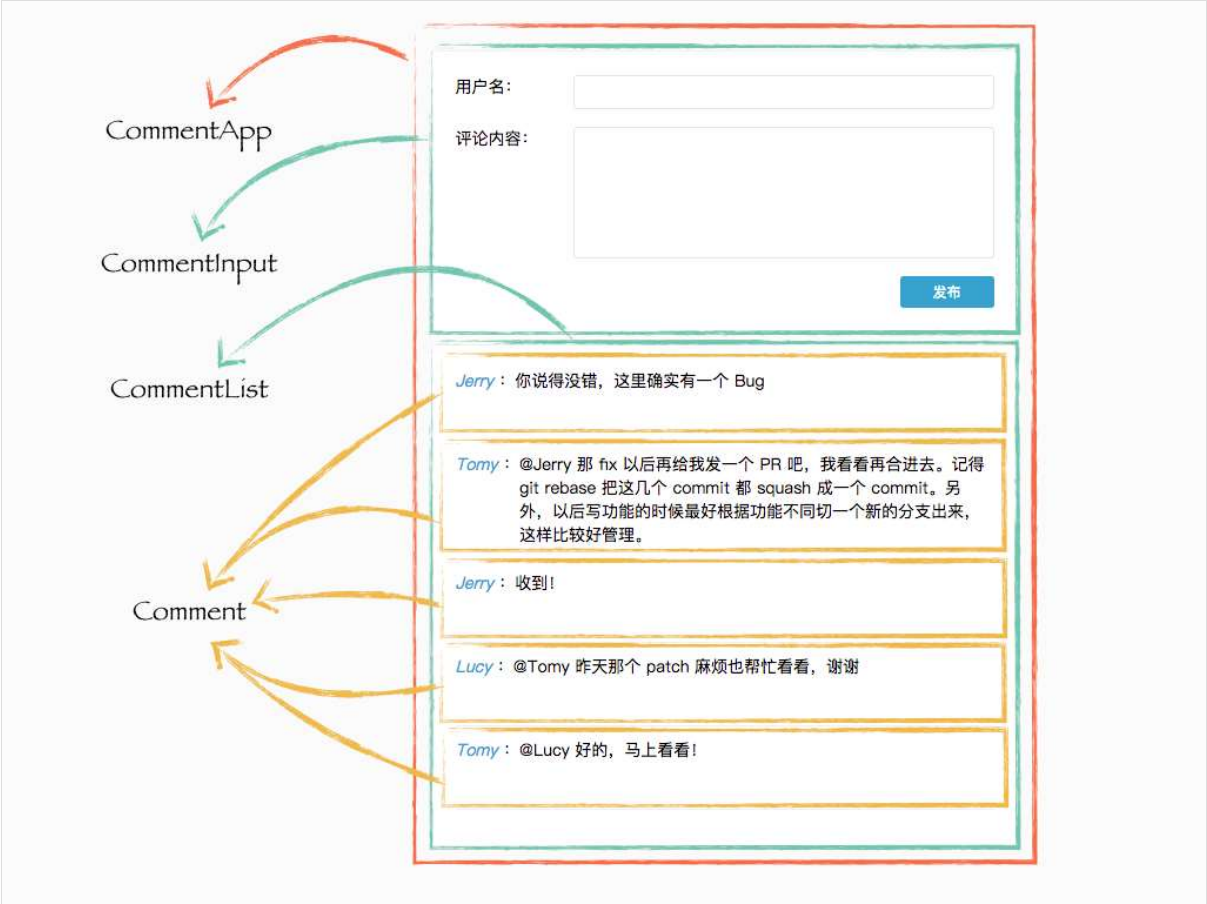
我们在上一阶段的评论功能基础上加上以下功能需求：

1. 页面加载完成自动聚焦到评论输入框。
2. 把用户名持久化，存放到浏览器的 LocalStorage 中。页面加载时会把用户名加载出来显示到输入框，用户就不需要重新输入用户名了。
3. 把已经发布的评论持久化，存放到浏览器的 LocalStorage 中。页面加载时会把已经保存的评论加载出来，显示到页面的评论列表上。
4. 评论显示发布日期，如“1 秒前”，”30 分钟前”，并且会每隔 5 秒更新发布日期。
5. 评论可以被删除。
6. 类似 Markdown 的行内代码块显示功能，用户输入的用 `` 包含起来的内容都会被处理成用 `<code>` 元素包含。例如输入 `` `console.log` `` 就会处理成 `<code>console.log</code>` 再显示到页面上。

在线演示地址。

大家可以在原来的第一阶段代码的基础上进行修改，第一、二阶段评论功能代码可以在这里找到： react-naive-book-examples。可以直接使用最新的样式文件 index.css 覆盖原来的 index.css。

接下来可以分析如何利用第二阶段的知识来构建这些功能，在这个过程里面可能会穿插一些小技巧，希望对大家有用。我们回顾一下这个页面的组成：



我们之前把页面分成了四种不同的组件：分别是 `CommentApp`、`CommentInput`、`CommentList`、`Comment`。我们开始修改这个组件，把上面的需求逐个完成。

自动聚焦到评论框

这个功能是很简单的，我们需要获取 `textarea` 的 DOM 元素然后调用 `focus()` API 就可以了。我们给输入框元素加上 `ref` 以便获取到 DOM 元素，修改 `src/CommentInput.js` 文件：

```
...
    <textarea
      ref={(textarea) => this.textarea = textarea}
      value={this.state.content}
      onChange={this.handleContentChange.bind(this)} />
...
```

组件挂载完以后完成以后就可以调用 `this.textarea.focus()`，给 `CommentInput` 组件加上 `ComponentDidMount` 生命周期：

```
class CommentInput extends Component {
  static propTypes = {
    onSubmit: PropTypes.func
  }

  constructor () {
```

```
    super()
    this.state = {
      username: '',
      content: ''
    }
  }

  componentDidMount () {
    this.textarea.focus()
  }
...
```

这个功能就完成了。现在体验还不是很好，接下来我们把用户名持久化一下，体验就会好很多。

大家可以注意到我们给原来的 `props.onSubmit` 参数加了组件参数验证，在这次实战案例中，我们都会给评论功能的组件加上 `propTypes` 进行参数验证，接下来就不累述。

持久化用户名

用户输入用户名，然后我们把用户名保存到浏览器的 LocalStorage 当中，当页面加载的时候再从 LocalStorage 把之前保存的用户名显示到用户名输入框当中。这样用户就不用每次都输入用户名了，并且评论框是自动聚焦的，用户的输入体验就好很多。

我们监听用户名输入框失去焦点的事件 `onBlur`：

```
...
    <input
      value={this.state.username}
      onBlur={this.handleUsernameBlur.bind(this)}
      onChange={this.handleUsernameChange.bind(this)} />
...
```

在 `handleUsernameBlur` 中我们把用户的输入内容保存到 LocalStorage 当中：

```
class CommentInput extends Component {
  constructor () {
    super()
    this.state = {
      username: '',
      content: ''
    }
  }

  componentDidMount () {
    this.textarea.focus()
  }
```

```
  _saveUsername (username) {
    localStorage.setItem('username', username)
  }

  handleUsernameBlur (event) {
    this._saveUsername(event.target.value)
  }
...
```

在 `handleUsernameBlur` 中我们把用户输入的内容传给了 `_saveUsername` 私有方法（所有私有方法都以 `_` 开头）。`_saveUsername` 会设置 LocalStorage 中的 `username` 字段，用户名就持久化了。这样就相当于每当用户输入完用户名以后（输入框失去焦点的时候），都会把用户名自动保存一次。

输入用户名，然后到浏览器里里面看看是否保存了：

然后我们组件挂载的时候把用户名加载出来。这是一种数据加载操作，我们说过，不依赖 DOM 操作的组件启动的操作都可以放在 `componentWillMount` 中进行，所以给 `CommentInput` 添加 `componentWillMount` 的组件生命周期：

```
...
  componentWillMount () {
    this._loadUsername()
  }

  _loadUsername () {
    const username = localStorage.getItem('username')
    if (username) {
      this.setState({ username })
    }
  }

  _saveUsername (username) {
    localStorage.setItem('username', username)
  }
...
```

`componentWillMount` 会调用 `_loadUsername` 私有方法，`_loadUsername` 会从 LocalStorage 加载用户名并且 `setState` 到组件的 `state.username` 中。那么组件在渲染的时候（`render` 方法）挂载的时候就可以用上用户名了。

这样体验就好多了，刷新页面，不需要输入用户名，并且自动聚焦到了输入框。我们 1、 2 需求都已经完成。

小贴士

这里插入一些小贴示，大家可以注意到我们组件的命名和方法的摆放顺序其实有一定的讲究，这里可以简单分享一下个人的习惯，仅供参考。

组件的私有方法都用 `_` 开头，所有事件监听的方法都用 `handle` 开头。把事件监听方法传给组件的时候，属性名用 `on` 开头。例如：

```
<CommentInput
  onSubmit={this.handleSubmitComment.bind(this)} />
```

这样统一规范处理事件命名会给我们带来语义化组件的好处，监听（`on`）`CommentInput` 的 `Submit` 事件，并且交给 `this` 去处理（`handle`）。这种规范在多人协作的时候也会非常方便。

另外，组件的内容编写顺序如下：

1. `static` 开头的类属性，如 `defaultProps`、`propTypes`。
2. 构造函数，`constructor`。
3. `getter/setter`（还不了解的同学可以暂时忽略）。
4. 组件生命周期。
5. `_` 开头的私有方法。
6. 事件监听方法，`handle*`。
7. `render*`开头的方法，有时候 `render()` 方法里面的内容会分开到不同函数里面进行，这些函数都以 `render*` 开头。
8. `render()` 方法。

如果所有的组件都按这种顺序来编写，那么维护起来就会方便很多，多人协作的时候别人理解代码也会一目了然。

> 因为第三方评论工具有问题，对本章节有任何疑问的朋友可以移步到 React.js 小书的论坛 发帖，我会回答大家的疑问。

下一节：实战分析：评论功能（五）

上一节：PropTypes 和组件参数验证

如果你觉得小书写得还不错，可以请胡子大哈喝杯茶 :)

赞赏

或者传播一下知识也是一个很好的选择

1 条评论，1 人参与。

7

我有话说...

使用社交帐号登录

发布前先点击左边的按钮登录

最新评论

最熟悉的陌生人 • 7月4日 16:01

加组件验证要加“ import PropTypes from 'prop-types‘ ”

顶 • 回复 • 分享»

友言?