25 底部输入框组件开发

更新时间：2019-09-16 10:36:36

> 人只有献身于社会，才能找出那短暂而有风险的生命的意义。
>
> ——爱因斯坦

我们在之前的朋友圈页面UI中，已经将输入框组件封装成了一个公共组件，那么接下来就要丰富这个组件的功能，让其为聊天页面服务，下面就进入开发吧。

本章节完整源代码地址，大家可以事先浏览一下：

Github-inputBar

Github-chat

图片上传逻辑

在聊天页面中，inputBar 组件还负责图片上传逻辑，首先需要添加在前端项目的 components 文件夹下 的inputBar文件夹的`index.vue`中，新增下面逻辑：

下面这段代码是用来初始化我们的 weui 图片上传uploader组件，需要注意的是放在`mounted`里是为了保证UI已经渲染完成，这样才能找到相关的dom节点进行初始化。

```
mounted () {

  let self = this
  weui.uploader('#uploader', {
    url: service.baseURL + 'post/uploadimgaliyun',
    auto: true,
    type: 'file',
    fileVal: 'image',
    compress: {
      width: 1300,
      height: 1300,
      quality: 0.8
    },
    onBeforeQueued: function (files) {
      // `this` 是轮询到的文件, `files` 是所有文件

      if (['image/jpg', 'image/jpeg', 'image/png', 'image/gif'].indexOf(this.type) < 0) {
        weui.alert('请上传图片')
        return false // 阻止文件添加
      }
      if (this.size > 10 * 1024 * 1024) {
        weui.alert('请上传不超过10M的图片')
        return false
      }
      if (files.length > 1) { // 防止一下子选择过多文件
        weui.alert('最多只能上传1张图片，请重新选择')
        return false
      }

      // return true; // 阻止默认行为，不插入预览图的框架
    },

    onBeforeSend: function (data, headers) {
      // console.log(this, data, headers);
      // $.extend(data, { test: 1 }); // 可以扩展此对象来控制上传参数
      // $.extend(headers, { Origin: 'http://127.0.0.1' }); // 可以扩展此对象来控制上传头部
      headers['wec-access-token'] = getCookie('token')
      // return false; // 阻止文件上传
    },

    onSuccess: function (ret) {


      //上传图片回调
      self.$emit('uploaded', ret)
      // return true; // 阻止默认行为，不使用默认的成功态
    }
  })
},
```

这里我相信大家应该很熟悉了，调用 weui 的图片上传组件来实现图片上传，这里要注意一下 uploader 的 id 和之前页面的不要重复，保证唯一。

在完成了 uploader 组件初始化后，在 inputBar 文件夹下的 index.vue 中编写图片上传面板的UI。

新增template代码：

```
<template>
  ...
  <div class="opera-panel" v-show="!option.noPlus">
    <div class="opera-item">
      <div class="item-icon" @click="upload"></div>
      <p class="item-text">照片</p>
      <div style="display:none;" id="uploader">
        <input ref="uploader" id="uploaderInput" class="weui-uploader__input" type="file" accept="image/*" />
      </div>
    </div>
  </div>
  ...
</template>
```

这里解释一下其中的一些逻辑：

1. option.noPlus 标志位是表明是否需要展示面板，在朋友圈页面，是不需要这个功能的，所以这里需要对应判断一下。
2. 将 uploader 设置成 display:none 的，来提供图片上传功能。

对应的css样式，这里之间采用 flex 的 column 布局即可：

```
.opera-panel {
  height: 230px;
  width: 100%;
  display: flex;
}
.opera-item {
  width:66px;
  height:90px;
  display: flex;
  justify-content: center;
  flex-direction: column;
  margin: 12px;
}
```

接下来，需要给在输入框的右侧新建一个加好按钮 .plus-btn，用来添加呼起图片上传面包逻辑：

```
<template>
  ...
   <div class="plus-btn" @click="showPanel" v-show="!option.noPlus"></div>
  ...
</template>
```

这里也需要 option.noPlus 这个标志位来判断，对应的JavaScript逻辑如下：

```
 /*
 * 打开图片操作面板
 */
 showPanel () {
   this.panelShow = true
   this.$emit('showBottom')
 },
 /*
 * 关闭图片操作面板
 */
 closePanel(){
   this.panelShow = false;
 },
```

对于输入框组件本身来说，我们通过调整父元素高度的方式来显示和隐藏图片操作面板：

```
.bottom-view {
  width: 100%;
  height: 56px;
  overflow: hidden;
  transition: height 200ms;
}
.bottom-view.show {
  width: 100%;
  height: 285px;
}
```

这个我们在前端项目的 views 文件夹下的chat文件夹的 index.vue 里编写逻辑，来控制隐藏和显示，代码如下：

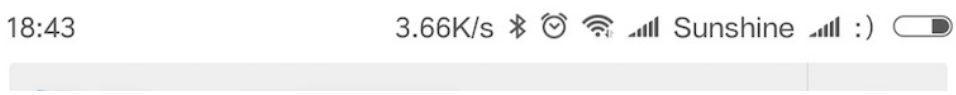
```
  /*
  * 显示图隐藏片操作面板
  */
  showBottom () {
    if (this.bottomClass.indexOf('show') > -1) {
      this.bottomStyle = ''
      // this.bottomClass = this.bottomClass.replace(' show','');
    } else {
      this.bottomClass += ' show'
    }
  },
```

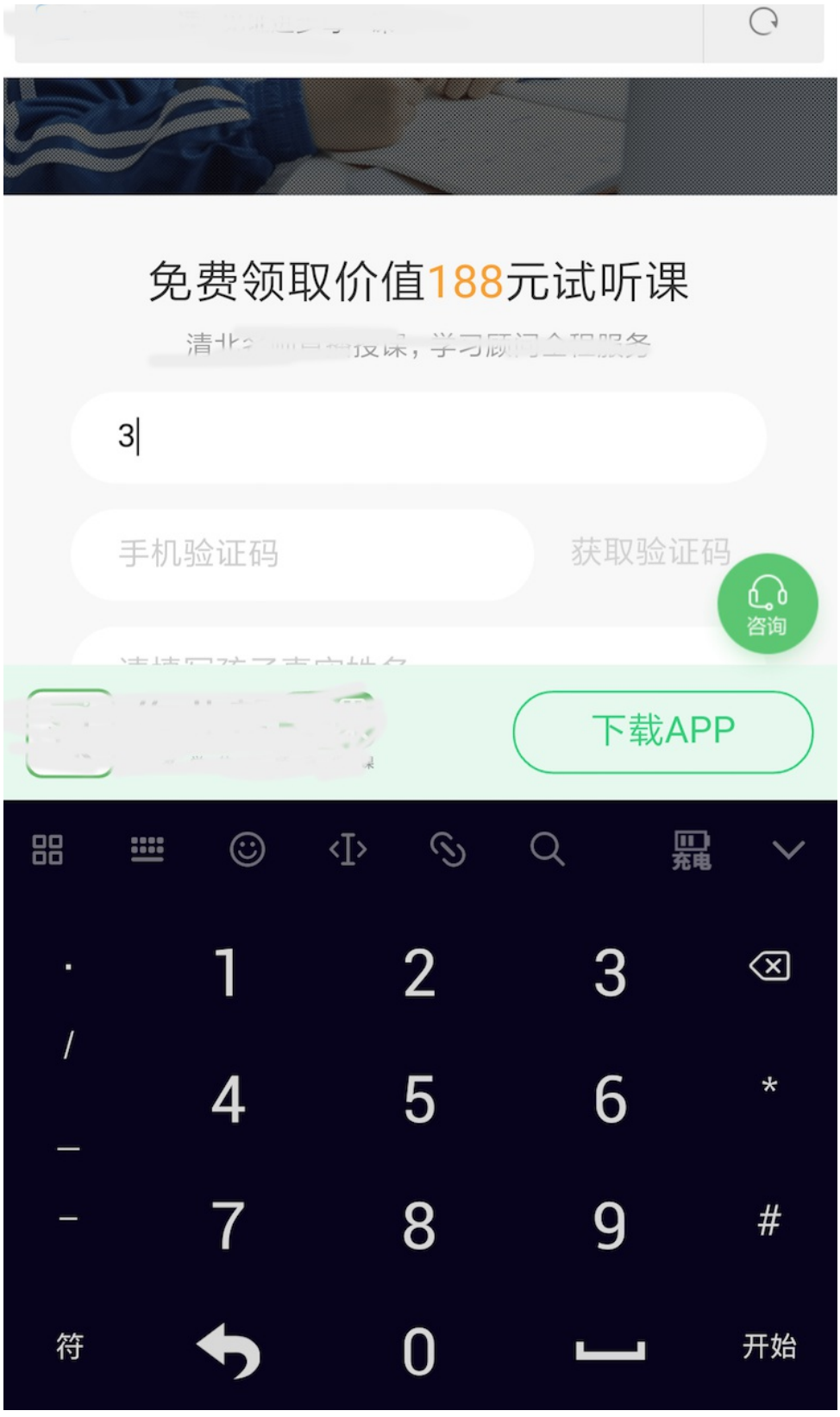
这里看起来没什么问题，但是真正我们在手机上测试时，会发现这样的结果：

当我们点开图片操作面板时，没什么问题，但是从图片面板状态直接进入输入时，会发现当键盘呼起时输入框被移动到了页面的最底部，在分析这个问题的原因时，我们先讲一个知识点：fixed定位元素在键盘呼起时的失效问题。

fixed定位元素在键盘呼起时的失效问题

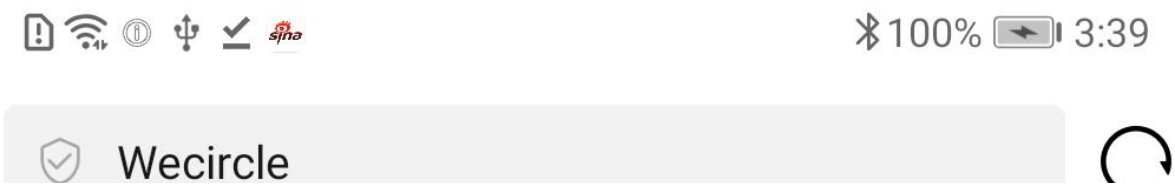
使用了 display:fixed;bottom:0; 并且在页面底部时，如果此时键盘呼起，这个元素会被顶上来，如下图：



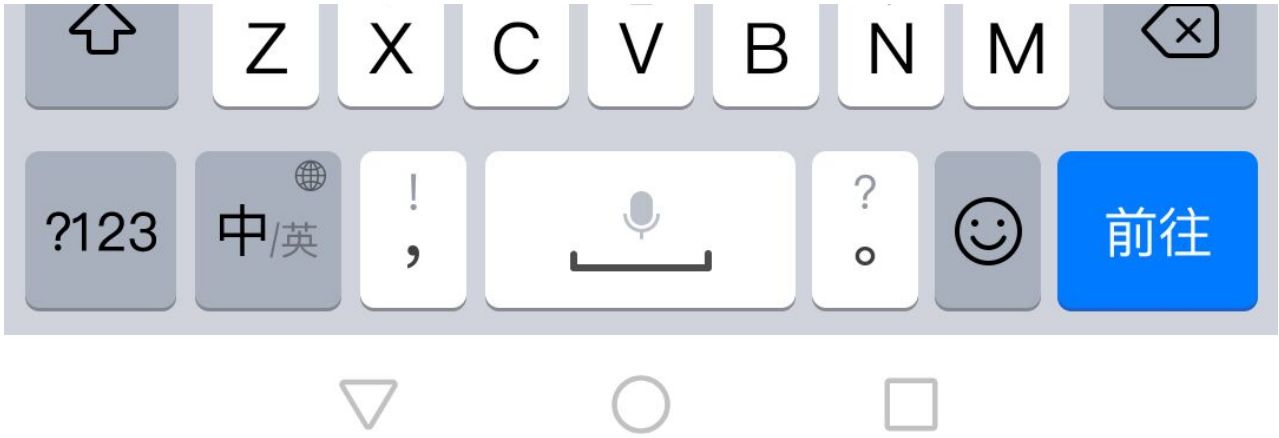


如果这个元素是 input，那么无论是否时 fixed 都会被顶上来，这个在Android和iOS上都会出现，只不过iOS上会把webview也顶上来，所以在iOS上，所有元素都会被顶上来，这个问题可以目前简单方法可以通过在键盘呼起时将 fixed 元素隐藏，键盘收回时在显示来解决。

而我们如果把输入框 input 放在底部，可以正好利用这个特性来实现我们输入框紧贴键盘的页面布局，如下图：



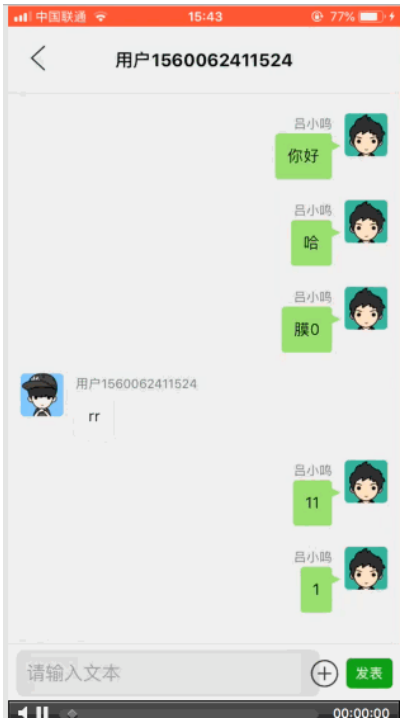

吕小鸣
请输入文本
发表
嗯 我 你 好 哦 在



好了下面就分析一下iOS上出现上面那个问题的原因：

1. 当图片操作面板展开时，我们的输入框其实并没有在底部，而是位于页面下半部分。
2. 此时输入框获得焦点，键盘被呼起，页面被顶起来，我们这时把高度减少，就会造成页面高度减少，输入框被挤下来。
3. 这个问题在android上并没有出现，原因是android并没有把页面顶起来，所以输入框始终时紧贴着键盘，无论我们怎么修改高度。

那么接下来，我们就在键盘呼起时不去修改高度，保持原样，然后看一下效果：



在重新呼起键盘时，输入框能看到了，由于我们没有修改高度，又导致输入框太高了，没有紧贴键盘，这里的原因是：

1. 当图片操作面板展开时，我们的输入框其实并没有在底部，而是位于页面下半部分。
2. 之前我们讲过，位于页面下半部分的输入框，在获取到焦点，键盘呼起时，webview会被顶上来（在iOS中），所以就会看到上图的效果。

在前端项目的 views 文件夹下的 chat 文件夹的 index.vue 里编写一个逻辑，代码如下：

借助我们之前的方案，页面被顶上来，我们在调用`window.scroll()`，将页面顶回去。

```
/*
* 在图片操作面板处于展开状态时的处理逻辑
*/
hideBottomOnPanel (h) {
 //将聊天界面滚动到底部，看到最新的聊天内容
 this.scrollToEnd(true)

 //此时图片操作面板处于展开状态时
 if (this.bottomClass.indexOf('show') > -1) {

  if (os.isIOS) {
   //将页面在顶回去
   window.scroll(0, 70) //键盘高度-图片操作面板高度+输入框高度

  } else {
   //Android无需修改，直接将图片操作面板隐藏即可
   this.closePanel()
  }


 }
}
```

最终效果：

小结

本章节主要讲解了聊天页面的底部输入框逻辑的开发。

相关技术点：

1. 移动端web端`fixed`定位元素在键盘呼起时的失效问题，以及如何规避和解决。
2. 移动端web端如果把输入框`input`放在底部，可以正好利用这个特性来实现我们输入框紧贴键盘的页面布局。

本章节完整源代码地址：

Github-inputBar

Github-chat

```
}
```

24 单条聊天内容组件

26 私信后端接口开发