22 用户资料页及后台接口开发

更新时间：2019-09-06 09:47:02

> 勤能补拙是良训，一分辛劳一分才。
>
> ——华罗庚

这一节我们主要进行我的用户资料页面 UI 和 后台接口开发，用户资料页面是指的查看其他用户的页面，可以用来查看名字，个性签名和电话等等数据，入口可以从朋友圈的单条内容组件的头像点击进来，逻辑相对简单。同时点击私信按钮可以进入到私信聊天界面，页面的主要元素是头像和昵称电话的展示。

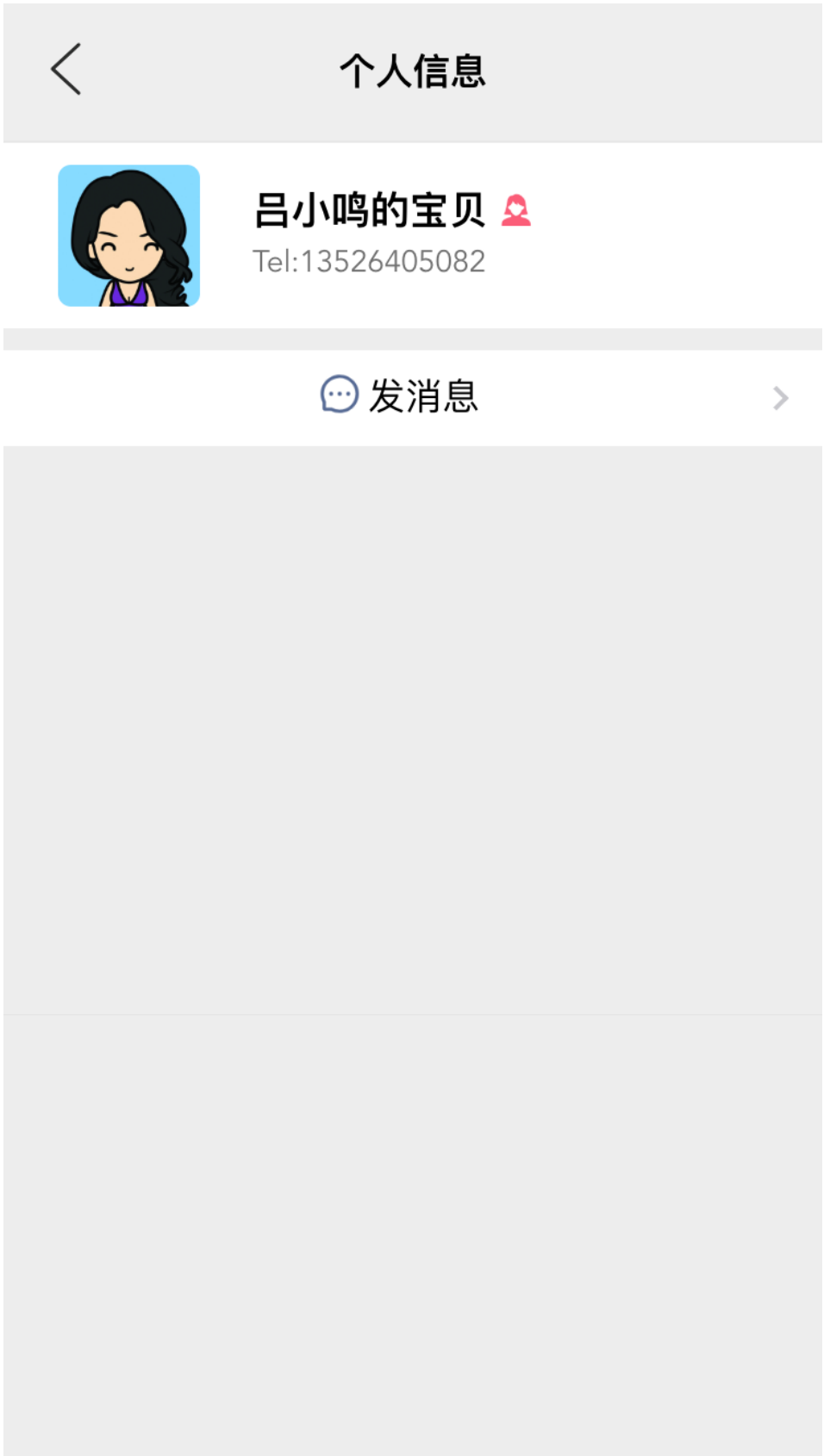
本章节完整源代码地址，大家可以事先浏览一下：

Github-personpage

Github-user.js

页面效果图

我们先来看一下，页面的 UI 效果图：



页面主要由顶部的 navHeader 和底部的个人信息元素和私信按钮组成：

页面逻辑开发

在前端项目的 views 文件夹下下新建 personpage 的文件夹，同时新建 `index.vue`：

```
<template>
  <div class="container">
    <navHeader title="个人信息"/>
    <div class="weui-cells content">
      <div class="panel person-info">
        <a class="weui-cell" href="javascript:;">
          <img class="avatar" :src="currentUser.avatar">
          <div class="person-info-right">
            <p :class="currentUser.gender == '1' ? 'male nickname' : 'female nickname' ">{{currentUser.nickname}}</p>
            <p class="phone">Tel:{{currentUser.phoneNum}}</p>
          </div>
        </a>
      </div>
      <div class="panel">
        <a class="weui-cell weui-cell_access" href="javascript:;" @click="goChat">
          <div class="weui-cell__bd">
            <div class="send-msg">
              <div class="msg-icon"></div>
              <span>发消息</span>
            </div>
          </div>
          <div class="weui-cell__ft"></div>
        </a>
      </div>
    </div>
  </div>
</template>
```

基本的 UI 使用`weui`的规范开发，其中：

1. 引入之前开发的 navHeader 组件。
2. 在判断用户性别时利用`:class="currentUser.gender == '1' ? 'male nickname' : 'female nickname' "`来给不同的性别以不同的class。
3. 展示性别的 icon 我们采用`after`伪类开发，代码如下：

```
.nickname {
  font-size: 18px;
  font-weight: bold;
  color: #000;
  position: relative;
  display: inline-block;
}

.nickname::after {
  content: "";
  display: block;
  width: 15px;
  height: 15px;
  position: absolute;
  background-size: cover;
  right: -20px;
  top: 4px;

}
.male::after {
  background-image: url("./img/man.png");
}
.female::after {
  background-image: url("./img/female.png");
}
```

代码中用到的 man.png 个 female.png 可以在github源码的素材库里面获取到。

使用伪类的好处是当昵称长度不固定时，可以让性别的icon紧跟在昵称后面。需要注意给 .nickname 设置 display:inline-block 。

在 create() 方法里面调用请求，通过url上的参数来获取用户的信息：

```
async created () {
  // 通过接口，获取当前用户的资料
  let resp = await service.get('users/userinfo', {
    userId: this.$route.query.id
  })

  if (resp.code === 0) {
    // 拿到数据赋值
    this.currentUser = resp.data
  }
},
```

点击私信按钮，进入私信页面：

```
goChat () {
  // 如果没有登录态 跳转登录
  if (!this.$store.state.currentUser._id) {
    this.$router.push({
      name: 'login'
    })
    return
  }
  this.$router.push({
    path: '/chat',

    query: {
      id: this.currentUser._id,
      name: this.currentUser.nickname
    }
  })
}
```

这里说明一下，我们用router传参时为何用 query 而不用 params :

1. query 要用 path 来引入， params 要用 name 来引入，接收参数都是类似的，分别是 this.$route.query.name 和 this.$route.params.name 。
2. query 更加类似于我们ajax中get传参， params 则类似于post，说的再简单一点，前者在浏览器地址栏中显示参数，后者则不显示 query 。
3. query 在页面刷新后参数不会丢失， params 则不行。这里也可以替换成 params 使用，但是个人觉得在看不见浏览器地址栏的场景下用 query ，看见则用 params 更加美观，同时参数过多可能超过地址栏的限制。

使用 params 传参：

```
this.$router.push({
  name: 'a',
  params: {
    key: 'key',
  }
})
```

使用 query 传参：

```
this.$router.push({
  path: '/a',
  query: {
    key: 'key',
  }
})
```

查看用户资料页面接口

在完成了前端页面开发之后，就需要开发后端对应的接口的。在后端项目的 routes 文件夹下，在 users.js 里新增一个路由。
下面这段代码主要是创建了一个 post 方法的路由，路径是 /userinfo，当浏览器请求 http://xx.xx.xx/userinfo 就会进入这个方法。

```
/*
* 根据id获取个人信息
*/
router.get('/userinfo', async (req, res, next)=> {

  try {
    var user = await User.findById(req.query.userId).exec();

    res.json({
      code:0,
      data:user
    });
  }catch(e){
    console.log(e)
    res.json({
      code:1,
      data:e
    });
  }

});
```

这段代码的主要功能是根据页面上传来的用户 ID 查询出当前的用户信息，然后返回给前段即可。其中 findById() 是mongoose提供的查询方法，通过_id字段查找单个文档，返回单个文档，类似的方法还有 findOne(), find() 等等。

- findOne(): 查找单个文档:第一个参数是筛选条件，第二个参数是需要查询的有哪些字段

```
MyModel.findOne({ type: 'iphone' }, 'name', { lean: true }, callback);
```

- find(): 查找多个文档:第一个参数是筛选条件，第二个参数需要查询哪些字段。

```
MyModel.find(id,callback);
```

更多方法可参考文档来开发。

小结

本章节主要讲解了用户资料页UI开发模块，包括了个人信息的展示，以及私信按钮的跳转逻辑。
相关知识点：

1. 在vue-router中，使用 query 的 params 区别。
2. 在编写性别icon时，使用CSS伪类的好处是当昵称长度不固定时，可以让性别的icon紧跟在昵称后面。

3. 使用 `mongoose` 提供的 `findById()` 方法来查询用户并返回给前端。

本章节完整源代码地址：

Github-personpage

Github-user.js

}

21 我的页面UI及后台接口开发

23 聊天容器组件开发