

1. 后台主页

1.1 获取用户的基本信息

1. 导入需要的脚本：

```
1      <!-- 导入 jQuery -->
2      <script src="/assets/lib/jquery.js"></script>
3      <!-- 导入自己封装的 baseAPI -->
4      <script src="/assets/js/baseAPI.js"></script>
5      <!-- 导入自己的 js 文件 -->
6      <script src="/assets/js/index.js"></script>
```

2. 定义 getUserInfo 函数：

```
1      // 获取用户的基本信息
2      function getUserInfo() {
3          $.ajax({
4              method: 'GET',
5              url: '/my/userinfo',
6              // headers 就是请求头配置对象
7              headers: {
8                  Authorization: localStorage.getItem('token') || ''
9              },
10             success: function(res) {
11                 if (res.status !== 0) {
12                     return layui.layer.msg('获取用户信息失败! ')
13                 }
14                 // 调用 renderAvatar 渲染用户的头像
15                 renderAvatar(res.data)
16             }
17         })
18     }
```

1.2 渲染用户头像

1. 定义 renderAvatar 函数：

```
1      // 渲染用户的头像
2      function renderAvatar(user) {
3          // 1. 获取用户的名称
4          var name = user.nickname || user.username
5          // 2. 设置欢迎的文本
6          $('#welcome').html('欢迎    ' + name)
7          // 3. 按需渲染用户的头像
8          if (user.user_pic !== null) {
9              // 3.1 渲染图片头像
10             $('.layui-nav-img')
11                 .attr('src', user.user_pic)
12                 .show()
13             $('.text-avatar').hide()
14         } else {
15             // 3.2 渲染文本头像
```

```

16     $('.layui-nav-img').hide()
17     var first = name[0].toUpperCase()
18     $('.text-avatar')
19         .html(first)
20         .show()
21     }
22 }

```

1.3 统一为有权限的接口设置headers请求头

1. 在 baseAPI 的 `ajaxPrefilter` 中添加如下代码：

```

1     // 统一为有权限的接口，设置 headers 请求头
2     if (options.url.indexOf('/my/') !== -1) {
3         options.headers = {
4             Authorization: localStorage.getItem('token') || ''
5         }
6     }

```

1.4 实现退出功能

1. 修改退出的 `<a>` 链接如下：

```

1     <a href="javascript:;" id="btnLogout"><span class="iconfont icon-tuichu">
</span>退出</a>

```

2. 实现退出功能：

```

1     var layer = layui.layer
2
3     // 点击按钮，实现退出功能
4     $('#btnLogout').on('click', function() {
5         // 提示用户是否确认退出
6         layer.confirm('确定退出登录?', { icon: 3, title: '提示' },
7             function(index) {
8                 //do something
9                 // 1. 清空本地存储中的 token
10                localStorage.removeItem('token')
11                // 2. 重新跳转到登录页面
12                location.href = '/login.html'
13
14                // 关闭 confirm 询问框
15                layer.close(index)
16            })
17     })

```

1.5 控制用户的访问权限

1. 在调用有权限接口的时候，指定 `complete` 回调函数：

```

1      // 不论成功还是失败，最终都会调用 complete 回调函数
2      complete: function(res) {
3          // console.log('执行了 complete 回调: ')
4          // console.log(res)
5          // 在 complete 回调函数中，可以使用 res.responseJSON 拿到服务器响应回来的
数据
6          if (res.responseJSON.status === 1 && res.responseJSON.message ===
'身份认证失败!') {
7              // 1. 强制清空 token
8              localStorage.removeItem('token')
9              // 2. 强制跳转到登录页面
10             location.href = '/login.html'
11         }
12     }

```

1.6 优化权限控制的代码

1. 将权限控制的代码，从每个请求中，抽离到 `ajaxPrefilter` 中：

```

1      // 注意：每次调用 $.get() 或 $.post() 或 $.ajax() 的时候，
2      // 会先调用 ajaxPrefilter 这个函数
3      // 在这个函数中，可以拿到我们给Ajax提供的配置对象
4      $.ajaxPrefilter(function(options) {
5          // 在发起真正的 Ajax 请求之前，统一拼接请求的根路径
6          options.url = 'http://ajax.frontend.itheima.net' + options.url
7
8          // 统一为有限权的接口，设置 headers 请求头
9          if (options.url.indexOf('/my/') !== -1) {
10             options.headers = {
11                 Authorization: localStorage.getItem('token') || ''
12             }
13         }
14
15         // 全局统一挂载 complete 回调函数
16         options.complete = function(res) {
17             // console.log('执行了 complete 回调: ')
18             // console.log(res)
19             // 在 complete 回调函数中，可以使用 res.responseJSON 拿到服务器响应回来的数据
20             if (res.responseJSON.status === 1 && res.responseJSON.message === '身
份认证失败!') {
21                 // 1. 强制清空 token
22                 localStorage.removeItem('token')
23                 // 2. 强制跳转到登录页面
24                 location.href = '/login.html'
25             }
26         }
27     })
28

```

2. 基本资料

2.1 创建基本资料对应的页面

1. 新建 `/user/user_info.html` 并初始化如下:

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.0"
6   />
7     <title>Document</title>
8     <!-- 导入 layui 的样式 -->
9     <link rel="stylesheet" href="/assets/lib/layui/css/layui.css" />
10    <!-- 导入自己的样式 -->
11    <link rel="stylesheet" href="/assets/css/user/user_info.css" />
12  </head>
13  <body>
14    <!-- 卡片区域 -->
15    <div class="layui-card">
16      <div class="layui-card-header">修改用户信息</div>
17      <div class="layui-card-body">
18        卡片式面板通常用于非白色背景色的主体内<br />
19        从而映衬出边框投影
20      </div>
21    </div>
22  </body>
23 </html>
```

2. 新建 `/assets/css/user/user_info.css` 并初始化如下:

```
1 html,
2 body {
3   margin: 0;
4   padding: 0;
5 }
6
7 body {
8   background-color: #f2f3f5;
9   padding: 15px;
10 }
```

3. 修改 `index.html` 中对应的 `<a>` 链接:

```
1 <a href="/user/user_info.html" target="fm"><i class="layui-icon layui-
2 icon-app"></i>基本资料</a>
```

2.2 绘制基本资料对应的表单

1. 编写如下的表单结构:

```
1 <!-- form 表单区域 -->
2 <form class="layui-form" action="">
3   <div class="layui-form-item">
4     <label class="layui-form-label">登录名称</label>
5     <div class="layui-input-block">
6       <input type="text" name="username" required lay-
7       verify="required" placeholder="请输入登录名称" autocomplete="off"
8       class="layui-input" readonly />
9     </div>
10  </div>
```

```

9         <div class="layui-form-item">
10             <label class="layui-form-label">用户昵称</label>
11             <div class="layui-input-block">
12                 <input type="text" name="nickname" required lay-
verify="required|nickname" placeholder="请输入用户昵称" autocomplete="off"
class="layui-input" />
13             </div>
14         </div>
15         <div class="layui-form-item">
16             <label class="layui-form-label">用户邮箱</label>
17             <div class="layui-input-block">
18                 <input type="text" name="email" required lay-
verify="required|email" placeholder="请输入用户邮箱" autocomplete="off"
class="layui-input" />
19             </div>
20         </div>
21         <div class="layui-form-item">
22             <div class="layui-input-block">
23                 <button class="layui-btn" lay-submit lay-filter="formDemo">
提交修改</button>
24                 <button type="reset" class="layui-btn layui-btn-primary">重
置</button>
25             </div>
26         </div>
27     </form>

```

2. 在页面底部导入如下的脚本：

```

1     <!-- 导入 layui 的 js -->
2     <script src="/assets/lib/layui/layui.all.js"></script>
3     <!-- 导入 jquery -->
4     <script src="/assets/lib/jquery.js"></script>
5     <!-- 导入自己的 js -->
6     <script src="/assets/js/user/user_info.js"></script>

```

3. 在 `user_info.js` 中编写如下的代码：

```

1     $(function() {
2         var form = layui.form
3
4         form.verify({
5             nickname: function(value) {
6                 if (value.length > 6) {
7                     return '昵称长度必须在 1 ~ 6 个字符之间！'
8                 }
9             }
10        })
11    })

```

2.3 获取用户的基本信息

1. 导入 `baseAPI`：

```

1     <script src="/assets/js/baseAPI.js"></script>

```

2. 在 `user_info.js` 中定义并调用 `initUserInfo` 函数：

```

1     initUserInfo()

```

```

2
3    // 初始化用户的基本信息
4    function initUserInfo() {
5        $.ajax({
6            method: 'GET',
7            url: '/my/userinfo',
8            success: function(res) {
9                if (res.status !== 0) {
10                    return layer.msg('获取用户信息失败!')
11                }
12                console.log(res)
13            }
14        })
15    }

```

2.4 使用form.val方法快速为表单赋值

1. 为表单指定 `lay-filter` 属性:

```

1    <form class="layui-form" lay-filter="formUserInfo"></form>

```

2. 调用 `form.val()` 方法为表单赋值:

```

1    form.val('formUserInfo', res.data)

```

3. 使用隐藏域保存用户的 `id` 值:

```

1    <!-- form 表单区域 -->
2    <form class="layui-form" lay-filter="formUserInfo">
3        <!-- 这是隐藏域 -->
4        <input type="hidden" name="id" value="" />
5
6        <!-- 省略其他代码 -->
7    </form>

```

2.5 实现表单的重置效果

1. 阻止表单的默认重置行为，再重新获取用户信息即可:

```

1    // 重置表单的数据
2    $('#btnReset').on('click', function(e) {
3        // 阻止表单的默认重置行为
4        e.preventDefault()
5        initUserInfo()
6    })

```

2.6 发起请求更新用户的信息

1. 阻止表单的默认提交行为，并发起数据请求:

```

1    // 监听表单的提交事件
2    $('.layui-form').on('submit', function(e) {
3        // 阻止表单的默认提交行为
4        e.preventDefault()
5        // 发起 ajax 数据请求
6        $.ajax({
7            method: 'POST',

```

```

8      url: '/my/userinfo',
9      data: $(this).serialize(),
10     success: function(res) {
11         if (res.status !== 0) {
12             return layer.msg('更新用户信息失败! ')
13         }
14         layer.msg('更新用户信息成功! ')
15         // 调用父页面中的方法，重新渲染用户的头像和用户的信息
16         window.parent.getUserInfo()
17     }
18 })
19 })

```

2. 注意：<iframe> 中的子页面，如果想要调用父页面中的方法，使用 window.parent 即可。

3. 重置密码

3.1 渲染重置密码的页面结构

1. 在 /user/user_pwd.html 页面中编写如下的结构：

```

1  <!DOCTYPE html>
2  <html lang="en">
3      <head>
4          <meta charset="UTF-8" />
5          <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6          <title>Document</title>
7          <link rel="stylesheet" href="/assets/lib/layui/css/layui.css" />
8          <link rel="stylesheet" href="/assets/css/user/user_pwd.css" />
9      </head>
10     <body>
11         <!-- 卡片区域 -->
12         <div class="layui-card">
13             <div class="layui-card-header">修改密码</div>
14             <div class="layui-card-body">
15                 <form class="layui-form">
16                     <div class="layui-form-item">
17                         <label class="layui-form-label">原密码</label>
18                         <div class="layui-input-block">
19                             <input type="password" name="oldPwd" required lay-
20 verify="required" placeholder="请输入原密码" autocomplete="off"
21 class="layui-input" />
22                         </div>
23                     </div>
24                     <div class="layui-form-item">
25                         <label class="layui-form-label">新密码</label>
26                         <div class="layui-input-block">
27                             <input type="password" name="newPwd" required lay-
28 verify="required" placeholder="请输入新密码" autocomplete="off"
29 class="layui-input" />
30                         </div>
31                     </div>
32                     <div class="layui-form-item">
33                         <label class="layui-form-label">确认新密码</label>

```

```

30         <div class="layui-input-block">
31             <input type="password" name="rePwd" required lay-
verify="required" placeholder="请再次确认密码" autocomplete="off"
class="layui-input" />
32         </div>
33     </div>
34     <div class="layui-form-item">
35         <div class="layui-input-block">
36             <button class="layui-btn" lay-submit lay-filter="formDemo">
修改密码</button>
37             <button type="reset" class="layui-btn layui-btn-primary">重
置</button>
38         </div>
39     </div>
40 </form>
41 </div>
42 </div>
43 </body>
44 </html>

```

2. 在 `/assets/css/user/user_pwd.css` 中编写如下的样式：

```

1  html,
2  body {
3      margin: 0;
4      padding: 0;
5  }
6
7  body {
8      padding: 15px;
9      background-color: #f2f3f5;
10 }
11
12 .layui-form {
13     width: 500px;
14 }

```

3.2 为密码框定义校验规则

1. 定义如下的三个校验规则：

```

1  $(function() {
2      var form = layui.form
3
4      form.verify({
5          pwd: [/^\s{6,12}$/, '密码必须6到12位，且不能出现空格'],
6          samePwd: function(value) {
7              if (value === $(' [name=oldPwd] ').val()) {
8                  return '新旧密码不能相同！'
9              }
10         },
11         rePwd: function(value) {
12             if (value !== $(' [name=newPwd] ').val()) {
13                 return '两次密码不一致！'
14             }
15         }
16     })

```


2. 在 body 结束标签之前导入如下的 `script` 标签：

```

1      <!-- 导入 layui 的 js -->
2      <script src="/assets/lib/layui/layui.all.js"></script>
3      <!-- 导入 jQuery -->
4      <script src="/assets/lib/jquery.js"></script>
5      <!-- 导入 baseAPI -->
6      <script src="/assets/js/baseAPI.js"></script>
7      <!-- 导入自己的 js -->
8      <script src="/assets/js/user/user_pwd.js"></script>

```

3. 为密码框分别添加对应的校验规则：

```

1      <form class="layui-form">
2          <div class="layui-form-item">
3              <label class="layui-form-label">原密码</label>
4              <div class="layui-input-block">
5                  <input type="password" name="oldPwd" required lay-
verify="required|pwd" placeholder="请输入原密码" autocomplete="off"
class="layui-input" />
6              </div>
7          </div>
8          <div class="layui-form-item">
9              <label class="layui-form-label">新密码</label>
10             <div class="layui-input-block">
11                 <input type="password" name="newPwd" required lay-
verify="required|pwd|samePwd" placeholder="请输入新密码" autocomplete="off"
class="layui-input" />
12             </div>
13         </div>
14         <div class="layui-form-item">
15             <label class="layui-form-label">确认新密码</label>
16             <div class="layui-input-block">
17                 <input type="password" name="rePwd" required lay-
verify="required|pwd|rePwd" placeholder="请再次确认密码" autocomplete="off"
class="layui-input" />
18             </div>
19         </div>
20         <div class="layui-form-item">
21             <div class="layui-input-block">
22                 <button class="layui-btn" lay-submit lay-filter="formDemo">修改密码
</button>
23                 <button type="reset" class="layui-btn layui-btn-primary">重置
</button>
24             </div>
25         </div>
26     </form>

```

3.3 发起请求实现重置密码的功能

```

1      $('layui-form').on('submit', function(e) {
2          e.preventDefault()
3          $.ajax({
4              method: 'POST',
5              url: '/my/updatepwd',

```

```

6      data: $(this).serialize(),
7      success: function(res) {
8          if (res.status !== 0) {
9              return layui.layer.msg('更新密码失败! ')
10         }
11         layui.layer.msg('更新密码成功! ')
12         // 重置表单
13         $('.layui-form')[0].reset()
14     }
15 })
16 })

```

4. 更换头像

4.1 初步渲染更换头像页面的结构

1. 创建 `/user/user_avatar.html` 页面:

```

1  <!DOCTYPE html>
2  <html lang="en">
3      <head>
4          <meta charset="UTF-8" />
5          <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6          <title>Document</title>
7          <link rel="stylesheet" href="/assets/lib/layui/css/layui.css" />
8          <link rel="stylesheet" href="/assets/css/user/user_avatar.css" />
9      </head>
10     <body>
11         <!-- 卡片区域 -->
12         <div class="layui-card">
13             <div class="layui-card-header">更换头像</div>
14             <div class="layui-card-body">
15                 卡片式面板通常用于非白色背景色的主体内<br />
16                 从而映衬出边框投影
17             </div>
18         </div>
19     </body>
20 </html>
21

```

2. 美化基本样式:

```

1    html,
2    body {
3        margin: 0;
4        padding: 0;
5    }
6
7    body {
8        padding: 15px;
9        background-color: #f2f3f5;
10   }
11

```

3. 修改 `index.html` 中对应链接的属性:

```

1    <a href="/user/user_avatar.html" target="fm"><i class="layui-icon layui-
    icon-app"></i>更换头像</a>

```

4.2 实现裁剪区域图片的替换

```

1    // 为文件选择框绑定 change 事件
2    $('#file').on('change', function(e) {
3        // 获取用户选择的文件
4        var fileList = e.target.files
5        if (fileList.length === 0) {
6            return layer.msg('请选择照片!')
7        }
8
9        // 1. 拿到用户选择的文件
10       var file = e.target.files[0]
11       // 2. 将文件, 转化为路径
12       var imgURL = URL.createObjectURL(file)
13       // 3. 重新初始化裁剪区域
14       $image
15         .cropper('destroy') // 销毁旧的裁剪区域
16         .attr('src', imgURL) // 重新设置图片路径
17         .cropper(options) // 重新初始化裁剪区域
18   })

```

4.3 将裁剪后的头像上传到服务器

```

1    // 为确定按钮, 绑定点击事件
2    $('#btnUpload').on('click', function() {
3        // 1. 要拿到用户裁剪之后的头像
4        var dataURL = $image
5          .cropper('getCroppedCanvas', {
6            // 创建一个 Canvas 画布
7            width: 100,
8            height: 100
9          })
10         .toDataURL('image/png') // 将 Canvas 画布上的内容, 转化为 base64 格式的字符串
11       // 2. 调用接口, 把头像上传到服务器
12       $.ajax({
13         method: 'POST',
14         url: '/my/update/avatar',
15         data: {
16           avatar: dataURL
17         }
18       })
19     })

```

```

17     },
18     success: function(res) {
19         if (res.status !== 0) {
20             return layer.msg('更换头像失败! ')
21         }
22         layer.msg('更换头像成功! ')
23         window.parent.getUserInfo()
24     }
25 })
26 })

```

4.4 设置头部区域的快捷方式

1. 打开 `index.html`，修改头部 `个人中心` 下的三个快捷方式如下：

```

1     <dl class="layui-nav-child">
2         <dd><a href="/user/user_info.html" target="fm">基本资料</a></dd>
3         <dd><a href="/user/user_avatar.html" target="fm">更换头像</a></dd>
4         <dd><a href="/user/user_pwd.html" target="fm">重置密码</a></dd>
5     </dl>

```

5. 文章分类

5.1 创建并显示文章分类页面

1. 创建 `/article/art_cate.html` 页面，并初始化如下的UI结构：

```

1     <!DOCTYPE html>
2     <html lang="en">
3         <head>
4             <meta charset="UTF-8" />
5             <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6             <title>Document</title>
7             <link rel="stylesheet" href="/assets/lib/layui/css/layui.css" />
8             <link rel="stylesheet" href="/assets/css/article/art_cate.css" />
9         </head>
10        <body>
11            <!-- 卡片区域 -->
12            <div class="layui-card">
13                <div class="layui-card-header">
14                    <span>文章类别管理</span>
15                    <button type="button" class="layui-btn layui-btn-normal layui-
16                        btn-sm">添加类别</button>
17                </div>
18                <div class="layui-card-body">
19                    <table class="layui-table">
20                        <colgroup>
21                            <col />
22                            <col />
23                            <col width="200" />
24                        </colgroup>
25                        <thead>

```

```

25         <tr>
26             <th>分类名称</th>
27             <th>分类别名</th>
28             <th>操作</th>
29         </tr>
30     </thead>
31     <tbody>
32         <tr>
33             <td>贤心</td>
34             <td>2016-11-29</td>
35             <td>人生就像是一场修行</td>
36         </tr>
37         <tr>
38             <td>许闲心</td>
39             <td>2016-11-28</td>
40             <td>于千万人之中遇见你所遇见的人，于千万年之中，时间的无涯的荒野里...
41         </td>
42     </tr>
43 </tbody>
44 </table>
45 </div>
46 </div>
47 </body>
48 </html>

```

2. 定义 `/assets/css/article/art_cate.css` 美化样式：

```

1  html,
2  body {
3      margin: 0;
4      padding: 0;
5  }
6
7  body {
8      padding: 15px;
9      background-color: #f2f3f5;
10 }
11
12 .layui-card-header {
13     display: flex;
14     justify-content: space-between;
15     align-items: center;
16 }
17

```

3. 修改 `index.html` 中对应的 `<a>` 链接：

```

1  <a href="/article/art_cate.html" target="fm"><i class="layui-icon layui-
    icon-app"></i>文章类别</a>

```

5.2 获取并使用模板引擎渲染表格的数据

1. 在页面底部导入模板引擎：

```

1  <script src="/assets/lib/template-web.js"></script>

```

2. 定义模板：

```

1      <!-- 表格数据的模板 -->
2      <script type="text/html" id="tpl-table">
3          {{each data}}
4          <tr>
5              <td>{{ $value.name }}</td>
6              <td>{{ $value.alias }}</td>
7              <td>
8                  <button type="button" class="layui-btn layui-btn-xs">编辑
9              <button type="button" class="layui-btn layui-btn-danger layui-
10                 btn-xs">删除</button>
11          </td>
12          </tr>
13          {{/each}}
14      </script>

```

- 发起请求获取数据：

```

1      initArtCateList()
2
3      // 获取文章分类的列表
4      function initArtCateList() {
5          $.ajax({
6              method: 'GET',
7              url: '/my/article/cates',
8              success: function(res) {
9                  var htmlStr = template('tpl-table', res)
10                 $('#tbody').html(htmlStr)
11             }
12         })
13     }

```

5.3 使用layer.open实现弹出层效果

- 导入 `layer`：

```

1      var layer = layui.layer

```

- 为按钮添加 `id` 属性：

```

1      <button type="button" class="layui-btn layui-btn-normal layui-btn-sm"
2          id="btnAddCate">添加类别</button>

```

- 在按钮的点击事件中，通过 `layer.open()` 展示弹出层：

```

1      // 为添加类别按钮绑定点击事件
2      $('#btnAddCate').on('click', function() {
3          layer.open({
4              type: 1,
5              area: ['500px', '250px'],
6              title: '添加文章分类',
7              content: 'ABC'
8          })
9      })

```

5.4 在弹出层中渲染form表单结构

1. 在页面中定义如下的 `script` 标签:

```
1      <script type="text/html" id="dialog-add">
2          <form class="layui-form" id="form-add">
3              <div class="layui-form-item">
4                  <label class="layui-form-label">分类名称</label>
5                  <div class="layui-input-block">
6                      <input type="text" name="name" required lay-
verify="required" placeholder="请输入分类名称" autocomplete="off"
class="layui-input">
7                  </div>
8              </div>
9              <div class="layui-form-item">
10                 <label class="layui-form-label">分类别名</label>
11                 <div class="layui-input-block">
12                     <input type="text" name="alias" required lay-
verify="required" placeholder="请输入分类别名" autocomplete="off"
class="layui-input">
13                 </div>
14             </div>
15             <div class="layui-form-item">
16                 <div class="layui-input-block">
17                     <button class="layui-btn" lay-submit lay-filter="formDemo">确
认添加</button>
18                     <button type="reset" class="layui-btn layui-btn-primary">重置
</button>
19                 </div>
20             </div>
21         </form>
22     </script>
```

2. 通过 `content` 属性指定内容:

```
1      layer.open({
2          type: 1,
3          area: ['500px', '250px'],
4          title: '添加文章分类',
5          content: $('#dialog-add').html()
6      })
```

5.5 实现添加文章分类的功能

1. 发起Ajax请求:

```
1      // 通过代理的形式, 为 form-add 表单绑定 submit 事件
2      $('body').on('submit', '#form-add', function(e) {
3          e.preventDefault()
4          $.ajax({
5              method: 'POST',
6              url: '/my/article/addcates',
7              data: $(this).serialize(),
8              success: function(res) {
9                  if (res.status !== 0) {
10                      return layer.msg('新增分类失败! ')
11                  }
12                  initArtCateList()
13                  layer.msg('新增分类成功! ')
14              }
15          })
16      })
```

```
14         // 根据索引，关闭对应的弹出层
15         layer.close(indexAdd)
16     }
17 })
18 })
```

2. 预先保存弹出层的索引:

```
1     // 为添加类别按钮绑定点击事件
2     var indexAdd = null
3     $('#btnAddCate').on('click', function() {
4         indexAdd = layer.open({
5             type: 1,
6             area: ['500px', '250px'],
7             title: '添加文章分类',
8             content: $('#dialog-add').html()
9         })
10    })
```