



云开发公开课

美食地图小程序



目录

Part 1: 云开发项目实战	2
1.1 项目概述	2
1.2 任务目标	2
1.3 准备工作	2
1.4 实战架构	3
1.5 实战任务	3
1.5.1 创建小程序·云开发环境	3
1.5.2 初始化云开发能力	5
1.5.3 开发首页的数据查询功能	6
1.5.4 开发首页的自动刷新功能	7
1.5.5 实现小程序的分享功能	7
1.5.6 开发列表功能	8
1.5.7 开发美食详情页面的功能	9
1.5.8 实现小程序新增数据的功能	11
1.5.9 实现小程序搜索的功能	13
1.5.10 实现管理员身份鉴权	14
1.5.11 配置上传，并部署云函数	16
1.6 完成标准	17
1.7 附加任务	17

Part 1：云开发项目实战

1.1 项目概述

地图相关的应用，是小程序结合线下场景中非常重要的一个工具，有了地图，小程序便完成了 Online to Offline 的转变，从一个纯线上的应用，变成了一个 O2O 应用。

本次「云开发」公开课，将通过实战“美食地图小程序”开发，教你如何截图小程序·云开发的能力，提升功能开发的效率。

1.2 任务目标

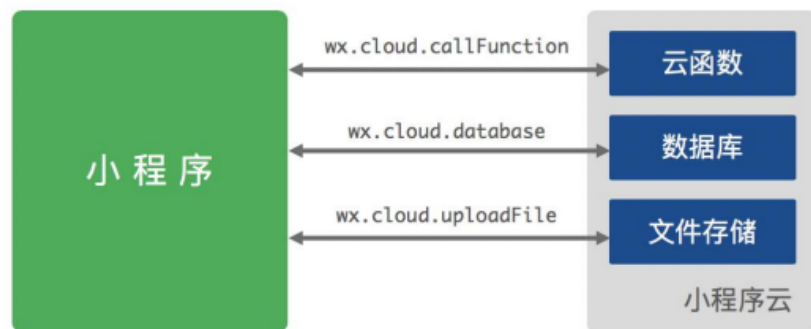
根据项目的初始框架，完善代码并使得查看地图、查看详情、新增美食等功能能够正常使用，完成美食地图小程序的开发，并上传设为体验版。并将助教微信号添加为体验者以便助教同学体验。

1.3 准备工作

在 PC 中下载美食地图小程序项目脚手架，下载地址：

<https://github.com/CloudKits/miniprogram-foodmap/archive/task-for-wechat-class-by-comment.zip>

1.4 实战架构



1.5 实战任务

***有能力的同学推荐尝试自行编写代码，不要偷看参考答案噢！**

1.5.1 创建小程序·云开发环境

1、打开微信 IDE，创建一个新的小程序项目，并选择导入美食地图 Demo 的目录，AppID 修改为你已经注册好的小程序 AppID。



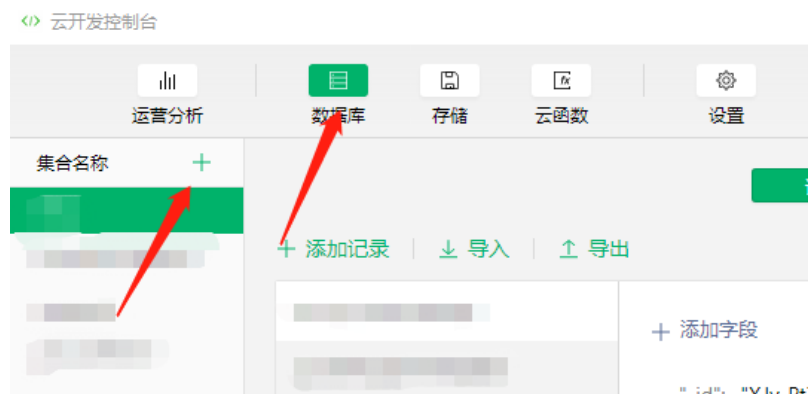
2、打开项目后，点击顶部工具栏上的“云开发”



3、自定义填入环境名称和环境 ID，点击“确定”创建云开发环境，即可进入云开发控制台。



4、美食地图小程序后续需要使用云开发提供的云数据库能力，数据库基于 MongoDB，需要先创建一个集合。进入云开发控制台，找到“数据库”，点击左侧边栏的“+”按钮创建集合。



5、我们需要创建两个集合，分别命名为“store”和“userInfo”，用于存储美食数据和个人数据。



需要注意的是，store 和 userInfo 两个集合的权限不同，store 的权限是**所有用户可读**，**仅创建者及管理员可写**；而 userInfo 的权限是**仅创建者及管理员可读写**。

1.5.2 初始化云开发能力

打开 miniprogram 目录下的 app.js 文件，在 onLaunch 事件中初始化云开发能力，其中，env 参数为上一任务中创建的环境 ID，traceUser 参数值为 true，并保存。

在其中找到代码输入点

```

3  onLaunch: function (options) {
4      /**
5       * 初始化云开发
6       */
7      if (!wx.cloud) {
8          console.error('请使用 2.2.3 或以上的基础库以使用云能力')
9      } else {
10         /**
11          * @task #1 初始化云开发能力
12          * @chapter 1.5.2 初始化云开发能力
13          * 请在下方输入代码
14          */
15
16
17     }

```

代码输入点

输入如下代码

```

wx.cloud.init({
  traceUser: true,
  env: config.envID
})

```

1.5.3 开发首页的数据查询功能

从数据库查询数据

打开 miniprogram/pages/map 目录下的 map.js 文件，在 onload 函数中取消云

开发数据库查询的代码的注释

```

35
36  /**
37   * @task #2 开发首页的数据查询功能
38   * @chapter 1.5.3 开发首页的数据查询功能
39   * 请取消下方代码的注释
40   */
41
42
43  // store.get().then(res => {
44  //   let data = res.data;
45  //   data.map(item => {
46  //     item.id = item._id
47  //   });
48  //   this.setData({
49  //     stores: res.data,
50  //     windowHeight: app.globalData.windowHeight,
51  //     hideMe:false,
52  //     showAdmin: showAdmin,
53  //     defaultScale: config.default_scale
54  //   }, () => {
55  //     wx.hideLoading();
56  //     wx.showToast({
57  //       title: '双指缩放可以调整地图可视区域，查看更多美食',
58  //       icon: 'none'
59  //     })
60  //   })
61  // })

```

取消此处代码前的注释



这段代码先使用 store.get 获取到了数据库中的 20 条数据，随后，在其 then 的 promise 回调中使用 map 方法进行了数据的更新，并使用了小程序提供的 setData 方法完成了数据的设置。

1.5.4 开发首页的自动刷新功能

在小程序中, `onLoad` 和 `onShow` 生命周期有所不同, `onLoad` 只在页面第一次加载时使用, 而 `onShow` 会在页面再一次显示时执行。

在美食地图中, 我们希望用户在进入到详情中返回到地图时, 可以自动刷新地图, 这样就可以实现用户看到的数据始终是最新的, 因此, 我们需要让页面每一次显示时, 都可以去加载一次数据, 这个功能用 `onShow` 生命周期函数就刚刚好了。

在这一次的任务中, 你需要做的是在小程序首页中实现这样的功能。你需要在 `miniprogram/pages/map` 目录下的 `map.js` 文件的 `onShow` 函数中取消如下代码的注释

```
66   onShow: function () {
67
68     /**
69      * @task #3 开发首页的自动刷新功能
70      * @chapter 1.5.4 实现小程序的自动刷新功能
71      * 请取消下方代码的注释
72      */
73
74     // store.get().then(res => {
75     //   let data = res.data;
76     //   data.map(item => {
77     //     item.id = item._id
78     //   });
79     //   this.setData({
80     //     stores: res.data
81     //   })
82     // })
83
84
85   },
```



1.5.5 实现小程序的分享功能

在小程序中, 分享功能是一大利器, 我们会希望自己的小程序可以分享给朋友、分享到群聊, 这样, 我们的小程序可以实现更加快速的生长与裂变, 在这节课, 我们就为小程序的

首页加上分享功能。

你需要在 miniprogram/pages/map 目录下的 map.js 文件中的 onShareAppMessage 方法中取消代码的注释。



```

142  /**
143  * 用户点击右上角分享
144  */
145  onShareAppMessage: function () {
146
147    /**
148     * @task #4 实现小程序的分享功能
149     * @chapter 1.5.5 实现小程序的分享功能
150     * 请取消下方代码的注释
151     */
152
153    // return {
154    //   title: '我在' + config.appName + '上发现了好吃的，你也看看吧！',
155    //   path: '/pages/map/map?_mta_ref_id=group',
156    //   imageUrl: "/images/share.jpg"
157    // }
158
159  },

```

将上面步骤做完，此时你的美食地图小程序就可以分享出去啦。你可以试着去修改一下

其中 title 部分的文字，去感受不同文字在分享时的情况。

1.5.6 开发列表功能

除了在地图模式下看美食以外，我们可能还希望看到其他形态的美食，比如，以列表的形态看数据，这样可以更加清晰明了的了解不同店铺的特色。

打开 miniprogram/pages/list 目录下的 list.js 文件，在其中的 loadData 方法中取消代码的注释

这段代码实现了云开发数据库的数据查询，和前面不同的是，为了支持触底自动加载，这段代码中使用了 skip 来完成数据的跳跃查询，你能否理解这个为什么这样写呢？

```

20   loadData: function() {
21
22     /**
23      * @task #5 开发列表功能
24      * @chapter 1.5.6 开发列表功能
25      * 请取消下方代码的注释
26      */
27
28     // store.skip(this.data.numbers).get().then(res => {
29     //   if (res.data.length == 0) {
30     //     wx.showToast({
31     //       title: '没有别的店铺了!',
32     //       icon: 'none'
33     //     });
34     //     return;
35     //   }
36     //   this.setData({
37     //     stores: this.data.stores.concat(res.data),
38     //     numbers: this.data.numbers + res.data.length
39     //   });
40     // })
41
42   },

```



1.5.7 开发美食详情页面的功能

1.5.7.1 查询数据

打开 miniprogram/pages/info 目录下的 info.js 文件，在 onload 函数中，取消相

关代码注释

```

21
22   /**
23    * @task #6 开发美食详情页的功能
24    * @chapter 1.5.7.1 查询数据
25    * 请取消下方代码的注释
26    */
27
28    // store.doc(options.id).get().then(res => {
29    //   if (config.dynamic_title){
30    //     wx.setNavigationBarTitle({
31    //       title: res.data.title,
32    //     });
33    //   }
34    //   let keywords_array = res.data.keywords.split(',').map(item => { return item.sp
35    //   let keywords = [].concat.apply([], keywords_array);
36    //   res.data.keywords = keywords
37    //   this.setData({
38    //     store: res.data,
39    //     is_administrator: app.globalData.is_administrator
40    //   }, res => {
41    //     wx.hideLoading();
42    //   })
43    // })
44
45

```



这段代码中包含了多个功能，我们来一一讲解一下。这段代码实现了判断是否开启了动态标题（页面标题是否根据店名进行变化），如果开启，就调用小程序官方的 API，来实现标题的变更。

```

if (config.dynamic_title){
  wx.setNavigationBarTitle({
    title: res.data.title,
  });
}

```

这段代码则是将关键词属性按照英文逗号和中文逗号进行两次切割，这样在页面中展示的关键词就是一个一个单独的，而不是一长串字符串，体验更好。

```

// 两次切割以适配中英文逗号
let keywords_array = res.data.keywords.split(',').map(item => {
  return item.split(', ')
})
// 将数组压平
let keywords = [].concat.apply([], keywords_array);
res.data.keywords = keywords

```

1.5.7.2 完成数据删除的功能

新增的数据如果需要删除怎么办？我们可以在详情页删除它，那么，就需要我们实现删除数据的功能

我们可以在 info.js 中找到 deleteItem 函数，在其中取消相关代码注释，来实现小程序调用云开发数据库接口，删除数据的功能。

```

111 |         if (res.confirm) {
112 |             /**
113 |              * @task #7 完成数据删除的功能
114 |              * @chapter 1.5.7.2 完成数据删除的功能
115 |              * 请取消下方代码的注释
116 |              */
117 |             // store.doc(this.data.store._id).remove().then(res => {
118 |             //   wx.showToast({
119 |             //     title: '删除成功',
120 |             //     icon: 'success',
121 |             //     success: res => {
122 |             //       wx.navigateBack({
123 |             //         delta: 2
124 |             //       })
125 |             //     }
126 |             //   })
127 |             // })
128 |             // }).catch(error => {
129 |             //   wx.showToast({
130 |             //     title: '删除失败! 请添加微信 ixiqin_com 排查问题',
131 |             //   })
132 |             // })
133 |         }

```

取消此处代码注释



1.5.8 实现小程序新增数据的功能

我们完成了数据的查看、数据的删除，接下来，我们来试着做一下新增数据，新增数据完成以后，我们就可以去看一看具体的效果了。

1.5.8.1 完成图片上传的功能

图片上传功能需要打开 miniprogram/pages/add 目录下的 add.js，找到其中的 uploadPhoto 方法，取消其中代码的注释。

```
139     uploadPhoto(filePath) {  
140  
141         /**  
142          * @task #8 完成图片上传功能  
143          * @chapter 1.5.8.1 完成图片上传功能  
144          * 请取消下方代码的注释  
145          */  
146  
147         // return wx.cloud.uploadFile({  
148         //   cloudPath: `${Date.now()}-${Math.floor(Math.random(0, 1) * 10000000)}.png`,  
149         //   filePath  
150         // })  
151     }  
152 }
```



这段代码实现了在小程序中上传文件到云开发文件存储中，同时，使用日期和随机数，确保生成的文件名是唯一的。

1.5.8.2 完成图片批量上传的功能


在开发过程中，我们不可能只允许用户上传一张图片，因此，我们可以借助 Promise 提供的方法，来实现多个图片文件的上传。在 add.js 文件中的 uploadImage 方法中取消如下代码的注释，来实现文件的批量上传。

```

108  /**
109   * @task #9 实现图片的批量上传
110   * @chapter 1.5.8.2 实现图片的批量上传
111   * 请取消下方代码的注释
112   */
113
114   // const uploadTask = items.map(item => this.uploadPhoto(item.src))
115
116   // Promise.all(uploadTask).then(result => {
117
118     //   let urls = [];
119     //   for (const file of result) {
120     //     urls.push(file.fileID);
121     //   }
122     //   this.setData({
123     //     images: urls
124     //   }, res => {
125     //     wx.hideLoading();
126     //     wx.showToast({ title: '上传图片成功', icon: 'success' })
127     //   })
128     // }).catch(() => {
129     //   wx.hideLoading()
130     //   wx.showToast({ title: '上传图片错误', icon: 'error' })
131     // })

```

取消此处代码注释



1.5.8.3 完成数据新增的功能

上传完图片，接下来就是做数据的新增的功能了。新增的功能需要在 `createItem` 方法中添加如下代码。这段代码调用了云开发的数据库的 API，来完成数据新增的功能。

```
64
65 // store.add({
66 //   data: {
67 //     ...value,
68 //     thumbs_up: 1,
69 //     iconPath: "/images/food.png",
70 //     longitude: this.data.longitude,
71 //     latitude: this.data.latitude,
72 //     label: {
73 //       content: value.title
74 //     },
75 //     images: this.data.images
76 //   }
77 // }).then(res => {
78 //   wx.hideLoading();
79 //   wx.showToast({
80 //     title: '创建成功! ',
81 //     icon: 'success',
82 //     success: res => {
83 //       wx.navigateBack({
84 //         })
85 //     }
86 //   })
87 // }).catch(error => {
88 //   console.error(error);
89 // })
```



取消此处代码的注释

1.5.9 实现小程序搜索的功能

在一个好的美食小程序中，搜索功能是一定需要的，因此，我们需要一个搜索功能，接下来我们就来实现他。

打开 miniprogram/pages/search 目录下的 search.js 文件,在其中的 loadData 函数中取消如下代码注释

```

35
36  /**
37   * @task #11 完成搜索功能
38   * @chapter 1.5.9 完成搜索功能
39   * 请取消下方代码的注释
40   */
41
42   // store.skip(this.data.numbers).where({
43   //   title: db.RegExp({
44   //     regexp: this.data.keywords,
45   //     options: 'i',
46   //   })
47   // }).get().then(res => {
48   //   /**
49   //    * 如果没有数据，就提示没有商户了，并返回。
50   //    */
51   //   if (res.data.length == 0) {
52   //     this.setData({
53   //       searched:true
54   //     })
55   //   }
56   //   this.setData({
57   //     stores: this.data.stores.concat(res.data),
58   //     numbers: this.data.numbers + res.data.length
59   //   });
60   // })

```

取消此处代码注释

1.5.10 实现管理员身份鉴权

我们希望小程序可以区分管理员和普通用户，那么我们就需要一个方法获取到管理员身份，这个方法我们可以借助云函数来实现。

1.5.10.1 获取用户 openID

首先，我们需要一个用户的唯一 ID，这个 ID 将会用来作为我们的用户身份的判断，我们可以使用云开发中云函数自带的 openID 来进行判断。

你可以打开 cloudfunctions/getUserOpenId 目录下的 index.js 文件，然后在其中取消代码注释：

```

6 // 云函数入口函数
7 exports.main = async (event, context) => {
8
9   /**
10    * @task #12 获取用户身份 ID
11    * @chapter 1.5.10.1 获取用户身份 ID
12    * 请取消下方代码的注释
13    */
14
15    // const wxContext = cloud.getWXContext()
16
17    // return {
18    //   openid: wxContext.OPENID
19    // }
20 }

```

取消此处代码注释

1.5.10.2 添加管理员身份校验函数

除了获取到用户的 `openid` 以外，我们还需要有一个函数，来帮助我们判断当前用户是否是管理员，我们可以通过另外一个函数 `checkUserAuth` 来完成。

你可以打开 `cloudfunctions/checkUserAuth` 目录下的 `index.js` 文件，然后在其中添加如下代码，完成自己的用户身份权限校验的代码。

```

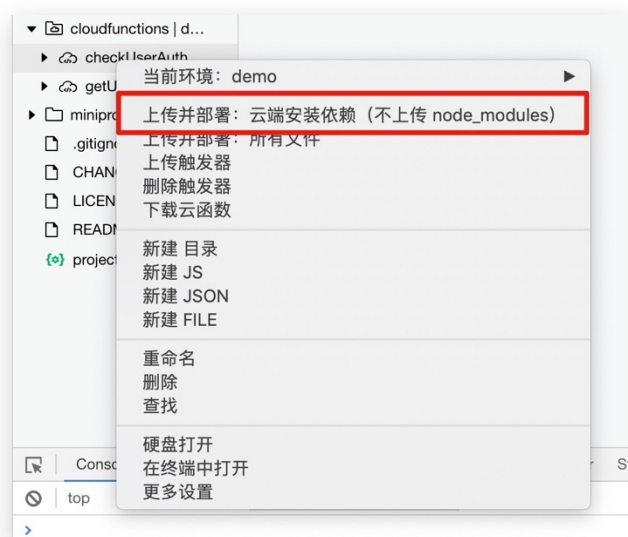
7 exports.main = async (event, context) => {
8   const wxContext = cloud.getWXContext()
9   const administrator = process.env.ADMIN.split('|');
10
11   /**
12    * @task #13 添加管理员身份校验函数 ID
13    * @chapter 1.5.10.2 添加管理员身份校验函数 ID
14    * 请取消下方代码的注释
15    */
16
17    // if (administrator.indexOf(wxContext.OPENID) == -1){
18    //   return {
19    //     data:{
20    //       is_administrator:false
21    //     }
22    //   }
23    // }else{
24    //   return {
25    //     data: {
26    //       is_administrator: true
27    //     }
28    //   }
29    // }

```

取消此处代码注释

1.5.11 配置上传，并部署云函数

在完成了小程序云函数的代码编写以后，就可以将两个函数进行部署了，部署时，你需要在函数上右击，选择「上传并部署：云端按照依赖（不上传 node_modules）」项目，来上传云函数。



并且，修改 app.json 和 config.js 中的配置项目，确保其中的配置项目都修改为你自己的。

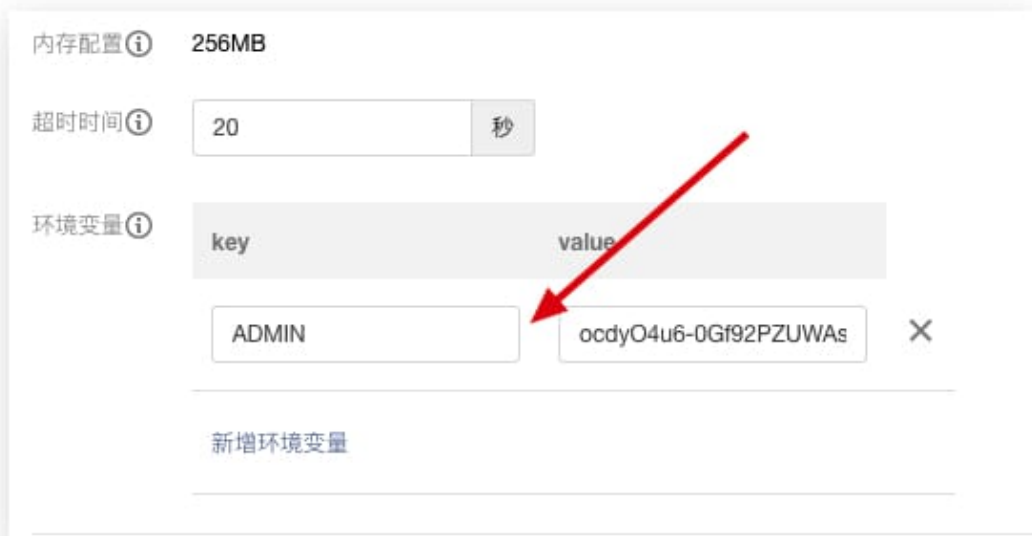
```

1  module.exports = {
2    ...
3    "appName": "深圳美食图鉴",
4    "envID": "demo-3ee737",
5    "mapSubKey": "URTBZ-FLY64-OVLUX-XHH4A-5DOGO-YKFFP",
6    "center_longitude": 113.921736,
7    "center_latitude": 22.538017,
8    "dynamic_title": true,
9    "show_admin": false,
10   "default_scale": 16
11 }

```

添加管理员

添加管理员可以通过在小程序中长按管理入口来获取到你的 openid，然后将你的 openid 添加到 checkUserAuth 云函数的环境变量 ADMIN 中。



1.6 完成标准

- 1、美食地图小程序可以添加数据、查看地图、查看列表等功能
- 2、所有数据需要借助云数据库能力存储在云端，并从云端拉取展示
- 3、所有图片类文件需要借助云存储能力存储在云端，并从云端拉取展示
- 4、任何小程序页面没有 Warning 报错

1.7 附加任务

如果你提前完成了云开发实战任务，可以思考一下，如何将美食地图小程序改为一个支持所有人上传数据的「大众点评」呢？



关注腾讯云「云开发」
上小程序，用云开发