小程序

微信公众平台 注册小程序 https://mp.weixin.qq.com/

小程序官方文档

https://developers.weixin.qq.com/miniprogram/dev/framework/

组件 API 工具

工具下载：https://developers.weixin.qq.com/miniprogram/dev/devtools/download.html

组件

view

text

button

```
//非常简单就可以实现微信客服的能力
<view>
  <text> hello world</text>
  <button type="primary" open-type="contact">客服</button>
</view>
```

camera

```
// html
<camera device-position="back" style="width: 100%; height: 300px;"></camera>
<button type="primary" bindtap="takePhoto">拍照</button>
<image mode="widthFix" src="{{src}}"></image>
```

```
//js
Page({

  /**
   * 页面的初始数据
   */
  //有点像vue中的data
  data: {
    src:""
  },
```

```
  takePhoto(){
    let ctx = wx.createCameraContext();
    ctx.takePhoto({
      success:res=>{
        console.log(res);
        //有点像react setState
        this.setData({
          src: res.tempImagePath
        })
      }
    })
  }

})
```

云开发

开发者可以使用云开发开发微信小程序、小游戏，无需搭建服务器，即可使用云端能力（Serverless 服务）。

`Serverless` = server + less (“少服务器的，或是无服务器的”)

云开发为开发者提供完整的原生云端支持和微信服务支持，弱化后端和运维概念，无需搭建服务器，使用平台提供的 API 进行核心业务开发，即可实现快速上线和迭代，同时这一能力，同开发者已经使用的云服务相互兼容，并不互斥。

传统开发模式

- 开发人员已经开发完毕->测试（测试环境）->(服务器，本地服务器或者阿里云服务器)-》上线（部署线上服务器）
- 我一个人想开发项目 从开发到上线 钱！！！

serverless

爬虫案例

https://book.douban.com/

- 支持isbn码搜索
- 我们借助小程序的扫码能力，获取图书的isbn码
- 借助云函数，实现对isbn对应的图书 信息抓取（对应之前的接口）

前端发送一个请求

后端给我数据

调整项目目录

project.config.json

```
"miniprogramRoot": "mp/", //小程序目录 把之前小程序的内容都放进来
"cloudfunctionRoot": "fn/",//云函数目录 指定的目录有特殊的图标
```

pages目录下新建book目录，初始化page

```
//book.wxml
<text>个人中心</text>
<button type="primary" bindtap="scanCode">添加图书</button>

//book.js
// mp/pages/book/book.js
Page({
  data: {},
  scanCode() {
    wx.scanCode({
      success: res => {
        console.log("isbn:" + res.result);
        //爬虫 小程序端没法做，我们需要云函数
      }
    });
  }
});
```

云函数写爬虫

- app.js

```
App({
  onLaunch: function() {
    //初始化云开发环境
    wx.cloud.init({
      //是否在将用户访问记录到用户管理中，在控制台中可见
      traceUser: true
    });
  }
});
```

- 右键fn目录新建云函数，其实就是nodejs项目 book14

```
// 云函数入口文件
const cloud = require("wx-server-sdk");
cloud.init();


// 云函数入口函数
// wx.cloud.callFunction的时候，就会执行这个main函数
exports.main = async (event, context) => {
  let { a, b } = event;
  return {
    sum: a + b
  };
};
```

- 上传并部署云函数 不然没有效果
- 小程序内调用云函数，使用wx.cloud.callFunction

```
// mp/pages/book/book.js
Page({
  data: {},
  scanCode() {
    wx.scanCode({
      success: res => {
        console.log("isbn:" + res.result);
        //爬虫 小程序端没法做，我们需要云函数
        wx.cloud.callFunction({
          name: "book14",//云函数名称
          data: {
            a: 10,
            b: 4,
            isbn:res.result
```

```
          },
          success: res => {
            console.log(res);
          }
        });
      }
    });
  }
});
```

开始爬虫逻辑

- book.js

```
wx.cloud.callFunction({
          name: "book14",//云函数名称
          data: {
            a: 10,
            b: 4,
            isbn:res.result  //把isbn传过去
          },
          success: res => {
            console.log(res);
          }
        });
```

- Book14 云函数
- 借助axios获取信息

```
npm i axios --save
```

```
// 云函数入口文件
const cloud = require("wx-server-sdk");
const axios = require("axios");//借助axios获取信息
cloud.init();

async function getDoubanBook(isbn) {
  //https://search.douban.com/book/subject_search?
search_text=9787229030933
  const url =
    "https://search.douban.com/book/subject_search?search_text=" + isbn;
  let res = await axios.get(url);
  console.log(res, res.data);
```

```
}
getDoubanBook("9787121331565");
// 云函数入口函数
// wx.cloud.callFunction的时候，就会执行这个main函数
exports.main = async (event, context) => {
  let { a, b, isbn } = event;
  return {
    sum: a + b
  };
};
```

- 借助doubanbook解密 拿到信息

```
npm i doubanbook --save
```

```
const doubanbook = require("doubanbook");




//用正则剥离加密的信息
  let reg = /window\.__DATA__ = "(.*)"/;
  if (reg.test(res.data)) {
    //第一个括号分组数据
    let searchData = doubanbook(RegExp.$1)[0];
    console.log(searchData);
    return searchData;
  }
```

```
//云函数 爬虫返回数据
...
exports.main = async (event, context) => {
  let { a, b, isbn } = event;
  let bookInfo = await getDoubanBook(isbn);
  return {
    sum: a + b,
    title: bookInfo.title
  };
};
```

- 上传并部署云函数
- wx.showToast 消息提示框 来增强体验
- Book.js

```
wx.cloud.callFunction({
          name: "book14",
          data: {
            a: 10,
            b: 4,
            isbn: res.result
          },
          success: res => {
            //消息提示框
            wx.showToast({
              title: res.result.title
            });
            //弹出对话框
            wx.showModal({
              title: res.result.title
            });
            console.log(res);
          }
        });
```

现在我们已经可以获取书籍的部分信息，其他大部分信息可以通过详情页来获取

我们可以通过cheerio 这个工具 通过jquery的语法来拿取数据

提供更多的信息

```
npm i cheerio --save
```

云函数 book14.js

```
const cheerio = require("cheerio");
...
exports.main = async (event, context) => {
  let { a, b, isbn } = event;
  let bookInfo = await getDoubanBook(isbn);
  //获取详情页的信息
  const detailPage = await axios.get(bookInfo.url);
  //使用cheerio 来解析这个页面,生成对象，可以使用jq语法
  const $ = cheerio.load(detailPage.data);
  //获取简介信息
  const summary = $("#link-report .intro").text();
  return {
    summary, //简介信息
    sum: a + b,
    title: bookInfo.title,
    image: bookInfo.cover_url, //图书的图片
```

```
    alt: bookInfo.url //详情页url
  };
};
```

入库操作

https://developers.weixin.qq.com/miniprogram/dev/wxcloud/guide/

数据库操作

增删改查

https://developers.weixin.qq.com/miniprogram/dev/wxcloud/guide/database/init.html

初始化

小程序book.js

```
//初始化 需要先获取数据库的引用
const db = wx.cloud.database();
page({})
```

插入新纪录

```
// mp/pages/book/book.js
const db = wx.cloud.database();
Page({
  data: {},
  scanCode() {
    wx.scanCode({
      success: res => {
        wx.showLoading();//显示loading
        wx.cloud.callFunction({
          ...
          success: res => {
            // 添加一个时间戳
            res.result.create_time = new Date().getTime();
            db.collection("books14").add({
              // data 字段表示需新增的 JSON 数据
              data: res.result,
              success: function(ret) {
                // res 是一个对象，其中有 _id 字段标记刚创建的记录的 id
```

```
                if (ret._id) {
                  console.log(ret);
                  wx.showModal({
                    title: "提示",
                    content: `${res.result.title}添加成功`
                  });
                }
                wx.hideLoading();//隐藏loading
              }
            });
          }
        });
      }
    });
  }
});
```

现在数据库里已经有数据里，

接下来我们使用数据在前端做展示操作

使用查询，比较复杂，会涉及到分页，排序，分组啊之类的。。。

新建books页面/pages操作

做一些页面标题设置

```
//books/app.json 做一些页面标题设置

{
  "usingComponents": {},
  "navigationBarTitleText": "图书列表"
}
```

获取数据，依然需要用到数据库

books.js

```
// mp/pages/books/books.js
const db = wx.cloud.database();
Page({
  /**
   * 页面的初始数据
   */
```

```
  data: {},

  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function(options) {
    //get 可以一次获取20条数据
    db.collection("books14").get({
      success: res => {
        // res.data 是一个包含集合中有权限访问的所有记录的数据，不超过 20 条
        this.setData({
          books: res.data
        });
      }
    });
  },

  /**
   * 页面上拉触底事件的处理函数
   */
  onReachBottom: function() {},
});
```

使用wx:for语法做列表展示

在组件上使用 `wx:for` 控制属性绑定一个数组，即可使用数组中各项的数据重复渲染该组件。

默认数组的当前项的下标变量名默认为 `index`，数组当前项的变量名默认为 `item`

列表渲染文档：https://developers.weixin.qq.com/miniprogram/dev/reference/wxml/

image mode文档：https://developers.weixin.qq.com/miniprogram/dev/component/image.html

```
<view wx:for="{{books}}">
  <view class="title">{{item.title}}</view>
  <image class="pic" mode="aspectFit" lazy-load="true" src="{{item.image}}">
</image>
  <text>{{item.summary}}</text>
</view>
```