# 笔记08

## 云开发

### 存储

#### 下载

download(e){

    // console.log(e)

    let  pic = e.target.dataset.pic; //获取图片的下载地址

    wx.showLoading({

      title:"图片下载中"

    })

    // 下载

    wx.cloud.downloadFile({

      fileID: pic, // 文件 ID

      success: res => {

        // 返回临时文件路径

        // console.log(res.tempFilePath)

        // 将下载下来的图片文件保存到系统相册内

        wx.saveImageToPhotosAlbum({

          filePath:res.tempFilePath,

          success(res) {

            // console.log(res)

            if(res.errMsg == "saveImageToPhotosAlbum:ok"){

              wx.showToast({

                title: '下载成功',

              })

              wx.hideLoading({

                complete: (res) => {},

              })

            }

           }

        })

      },

      fail: console.error

    })

#### 删除

  del(e) {

    let id = e.target.dataset.id;

    let pic = e.target.dataset.pic;

    let \_this = this;

    db.collection('user').doc(id).remove({

      success: (res) => {

        // console.log(res)

        if (res.errMsg == "document.remove:ok") {

          // 数据库删除成功了，把存储空间内的数据也要删除成功

          wx.cloud.deleteFile({

            fileList: [pic],  // 文件存储的id

            success: res => {

              // console.log(res.fileList)

              \_this.find()

            },

            fail: console.error

          })

        }

      }

    })

  },

## mpvue小程序开发框架

### 介绍

是由美团开发的！！！

mpvue （[github 地址请参见](https://github.com/Meituan-Dianping/mpvue" \t "http://mpvue.com/_blank) ）是一个使用 [Vue.js](https://vuejs.org" \t "http://mpvue.com/_blank) 开发小程序的前端框架。框架基于 Vue.js 核心，mpvue 修改了 Vue.js 的 [runtime](http://mpvue.com/mpvue) 和 [compiler](http://mpvue.com/mpvue-template-compiler) 实现，使其可以运行在小程序环境中，从而为小程序开发引入了整套 Vue.js 开发体验。

mpvue 继承自 Vue.js，其技术规范和语法特点与 Vue.js 保持一致

### 主要特性

* 彻底的组件化开发能力：提高代码复用性
* 完整的 Vue.js 开发体验
* 方便的 Vuex 数据管理方案：方便构建复杂应用
* 快捷的 webpack 构建机制：自定义构建策略、开发阶段 hotReload
* 支持使用 npm 外部依赖
* 使用 Vue.js 命令行工具 vue-cli 快速初始化项目
* H5 代码转换编译成小程序目标代码的能力

### 开发支持

文档： vue。Js文档，小程序开发文档， mpvue文档

编辑器： vscode,webstorm,sublime ........ (编写代码)

微信开发者工具：使用模拟器看效果，使用调试去调试

.vue文件在微信开发者工具在不认。

Nodejs，npm

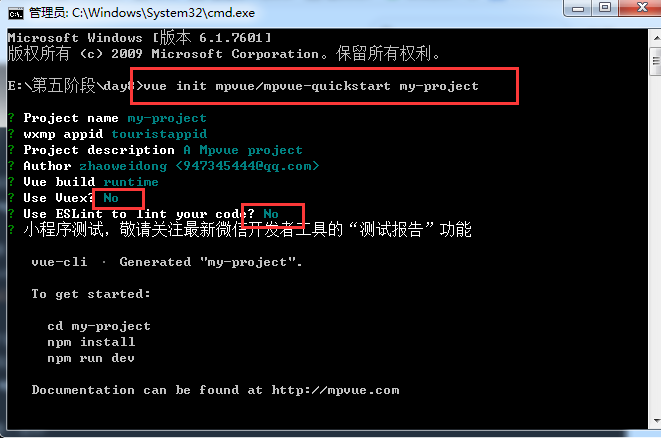
### 文档内容

<http://mpvue.com/mpvue/>

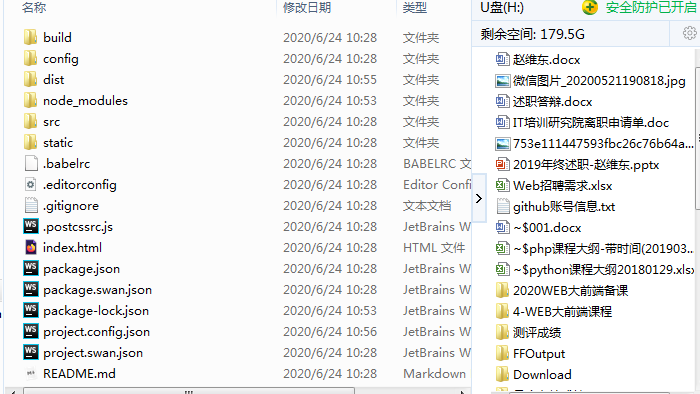
### mpvue项目创建 \*\*\*

现代前端开发框架和环境都是需要 Node.js 的，如果没有的话，请先下载 [nodejs](https://nodejs.org/" \t "http://mpvue.com/mpvue/_blank) 并安装。





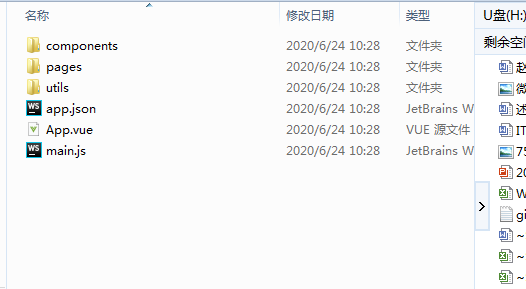
### 目录结构



#### Project.Config.json同小程序中的文件作用一致



#### Src目录：



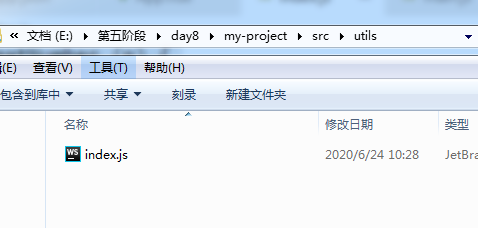
#### App.json 当前小程序的全局配置文件，与小程序中app.json作用保持一致



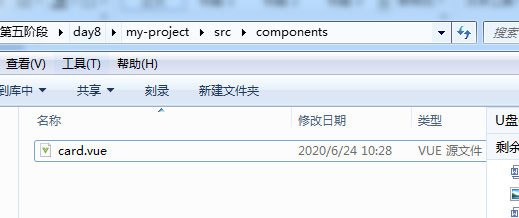
#### App.vue 全局的根组件



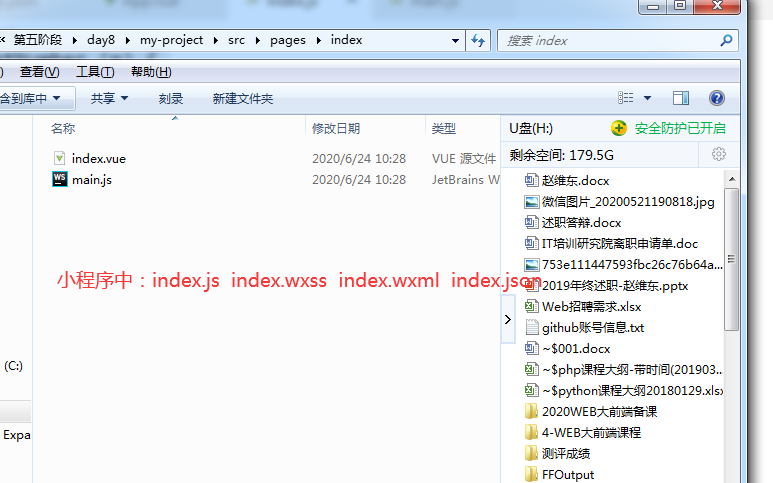
#### Utils 与微信小程序中utils保持一致



#### Components 存在自定义组件的目录



#### Pages页面目录



注意：

在页面的.vue的组件中：

<template>

  <div>

    <!-- html页面模板  -  wxml -->

    index

  </div>

</template>

<script>

//  相当于是页面的js

export default {

  data () {

    return {

    }

  },

  onShow(){

    console.log('onshow')

  }

}

</script>

<style scoped>

  /\* wxss 文件 \*/

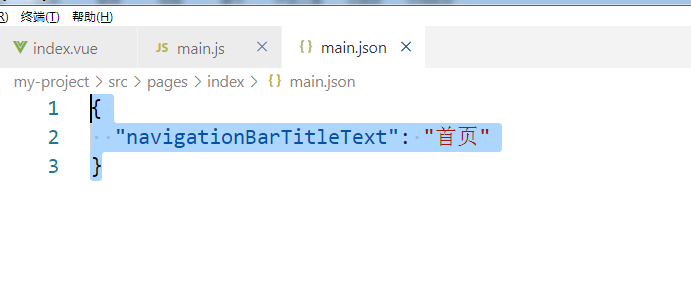
  div{

    color:blue

  }

</style>

在每一个页面中，都会有一个main.json文件，此文件就是当前页面的配置文件



每个页面的main.js:



注意：

在app，json中的pages属性上，后面必须改成main



### 创建页面

**当新创建页面的时候，一定要重新运行当前项目，否则页面加载不出来**

<template>

   <!--  页面结构  wxml -->

  <div class="app">

        我是test文件

        <!--  解析data内的数据变量 -->

        <div>{{ username }}</div>

        <!--  列表渲染  小程序： wx:for   vue:v-for  -->

        <!-- 小程序： wx:for  在mpvue中不可用 -->

        <!--<view  wx:for="{{ info }}">

            {{ item }}

        </view> -->

        <view>view组件</view>

         <!-- vue:v-for  在mpvue中可用 ,属性绑定也要使用vue里面绑定方式 -->

        <div v-for="(item,index) in  info" v-bind:key="index">

            {{ item }}

        </div>

        <!-- 条件渲染： 不能使用wx:if  要使用v-if -->

        <div v-if="onOff">

            我是真的

        </div>

        <div v-else>

            我是假的

        </div>

        <!--

        <image

          src="/static/images/user.png"

        >

        </image>

        <img src="/static/images/user.png"></img>

        -->

        <!-- 事件操作 -->

       <!-- <button size="mini" type="primary" bindtap="click"> 这个不能触发事件 -->

        <button size="mini" type="primary"  @click ="click(12,$event)" data-name="root">

       <!-- <button size="mini" type="primary"  @click ="click" data-id="100" data-name="admin"> -->

            点击事件

        </button>

  </div>

</template>

<script>

export default {

  name: '',

  components: {},

  data() {

    return {

        username:"百里守约",

        info:[10,20,30,40],

        onOff:false

    }

  },

  methods: {

    //   自定义的函数的编写/与小程序不同

    click(id,e){

        console.log('click',id,e)

    }

  },

}

</script>

<style lang="css" scoped>

</style>

## 项目

### 前台

首页、购物车页面、个人中心页面、商品列表页、商品详情页 mpvue

### 后台

Node + mongodb + express + ejs

商品分类管理、商品管理、订单管理、购物车管理、【banner 、商品属性、订单详情、优惠、会员、、、、、、】

### 创建项目

#### 3.1安装express框架

1. 使用npm init --yes 生成package.json文件
2. Cd进入项目根目录，使用npm install express --save
3. 检测是否安装成功！！

// 整个后台项目的入口文件

// 1.express

// 1-1. 引入express模块

const   express =  require('express');

// 1-2. 获取app对象

const  app   =  express()   //  new  express()

// 1-3. 设置当前服务的端口号

const  port = 3000;

// 1-5 设置路由，进行访问

app.get('/',(req,res)=>{

    res.send('1234567890')

})

// 1-4.监听

app.listen(port,()=>{

    console.log(`您的服务正在${port}端口运行！`)

})

1. supervisor 模块安装（热更新）

npm install -g supervisor

#### 3.2。安装ejs模块

1. 安装

Npm install [ejs@2.7.1](mailto:ejs@2.7.1) --save

1. 在app.js中设置模板引擎

App.set(‘view engine’ ,’ejs’)

1. 在项目的根目录下面创建views文件夹，用来存储所有的页面
2. 在路由中使用res.render()去加载页面

app.get('/',(req,res)=>{

    // res.send('9876')

    // render('页面路径',{传递参数})

    let  info = [

        {

            id:1001,

            name:"关羽"

        },

        {

            id:1002,

            name:"张飞"

        },

        {

            id:1003,

            name:"刘备"

        }

    ]

    res.render('test',{

        name:"李四",

        info,

    })

})

<!doctype html>

<html>

    <head>

        <meta  charset="utf-8"/>

        <title>ejs</title>

    </head>

<body>

    <!-- test -->

    <div>

        name:<%=  name %>

    </div>

    <div>

        <% for(let i  in  info){ %>

            id:<%= info[i].id %>  --name:<%= info[i].name %>

        <%}%>

    </div>

</body>

</html>

#### 3.3设置全局中间件

1. 处理样式问题

// 3.设置全局中间件

app.use('/',express.static('public'))

#### 3.4安装数据库模块

1. npm install mongodb@2.2.33 --save-dev
2. 在config.js中导入相关内容

// 4.引入数据库模块

const   MC  = require('mongodb').MongoClient;  //用来链接数据库

const   DBurl = "mongodb://localhost:27017/web" // 设置链接数据库的地址

### 商品分类模块

集合： type

字段：

\_id: 自动生成的唯一的主键id

Typename：类别名称

Typepic：类别图标

#### 获取文件上传信息

npm install multiparty

// 5.引入加载post请求及文件上传模块

var  multiparty  = require('multiparty');

// 执行添加

app.post('/type/doAdd',(req,res)=>{

    // res.send('1234567890')

    // 获取当前post传递过来的值  涉及到文件上传

    var form = new multiparty.Form();

    // 指定上传的文件目录  upload

    form.uploadDir = "upload";

    form.parse(req, function(err, fields, files) {

        //  fields  获取的是普通的post上传的内容 typename

        // files  获取的是文件上传的信息  typepic

        // console.log(fields,'---------------/////',files)

        let typename = fields.typename[0]; //获取到类别的名称

        let typepic = files.typepic[0].path;// 获取文件的存储路径

        let  data = {typename,typepic}

        // 执行数据库添加操作

        MC.connect(DBurl,(err,db)=>{

            // db 数据库引用

            // 连接表进行数据的添加

            db.collection('type').insertOne(data,(error,result)=>{

                // console.log(result)

                if(result.insertedId){

                    // 添加成功

                    res.send("<script>alert('类别添加成功!!!'); location.href='/type/index'; </script>")

                }else{

                    // 添加失败

                    res.send("<script>alert('类别添加失败!!!'); history.back(); </script>")

                }

            })

        })

    });

})

作业：

1. 童话故事，继续上线；
2. Mpvue的项目搭建起来，完成老师目前完成的效果；
3. 拓展，
   1. 把整个type模块自己写完