**一、 mpvue简介**

mpvue:是由 美团点评团队出品的小程序开 发的一款基于 vue的框架， 从整个 Vue的核心代码上经过二次开发而形成的一个框架，相当于是给Vue本身赋能，增加了开发微信小程序的能力。

使用 mpvue开发小程序，你将在小程序技术体系的基础上获取到这样一些能力：

l 彻底的组件化开发能力：提高代码

l 完整的 Vue.js 开发体验

l 方便的 Vuex 数据管理方案：方便构建复杂应用

l 快捷的 webpack 构建机制：自定义构建策略、开发阶段 hotReload

l 支持使用 npm 外部依赖

l 使用 Vue.js 命令行工具 vue-cli 快速初始化项目

l H5 代码转换编译成小程序目标代码的能力

它的目标是：在未来最理想的状态下，可以一套代码可以直接跑在多端： WEB、微信小程序、支付宝小程序、Native（借助weex）。不过由于各个端之间都存在一些比较明显的差异性，从产品的层面上讲，不建议这么做，这个框架的官方他们对它的期望的也只是开发和调试体验的一致

原生微信小程序、 mpvue、WePY这三种开发小程序方式的比较

**二、mpvue开发流程**

**1、小程序账号配置**

1）前往<https://mp.weixin.qq.com/wxopen/waregister?action=step1>根据指引填写信息和提交相应的资料，申请小程序帐号。在菜单 “设置”-“开发设置”获取小程序的 AppID 。

2）在菜单 “设置”-“开发设置”中配置服务器域名，必须是https开头的域名

**2、安装开发工具**

前往 开发者工具下载页面 ，根据自己的操作系统下载对应的安装包进行安装，有关开发者工具更详细的介绍可以查看 《开发者工具介绍》 。

打开小程序开发者工具，用微信扫码登录开发者工具，准备开发小程序。

**3、mpvue生成项目**

# 全局安装 vue-cli

$ npm install --global vue-cli

# 创建一个基于 mpvue-quickstart 模板的新项目

$ vue init mpvue/mpvue-quickstart my-project

# 安装依赖

$ cd my-project

$ npm install

# 启动构建

$ npm run dev

Npm run dev运行成功后，本地目录下会生成一个dist文件，这个文件就是生成的小程序相关代码。

在小程序中新建项目，填写上一步获取的 appid，便于后面可以在手机上预览，真机测试，小程序的文件目录就是本地项目目录的dist文件。

**三、mpvue开发中的规范**

**1、生命周期函数**

除了 vue本身的生命周期外，mpvue还兼容了小程序的生命周期，

app 部分：

onLaunch，初始化

onShow，当小程序启动，或从后台进入前台显示

onHide，当小程序从前台进入后台

page 部分：

onLoad，监听页面加载

onShow，监听页面显示

onReady，监听页面初次渲染完成

onHide，监听页面隐藏

onUnload，监听页面卸载

onPullDownRefresh，监听用户下拉动作

onReachBottom，页面上拉触底事件的处理函数

onShareAppMessage，用户点击右上角分享

onPageScroll，页面滚动

onTabItemTap, 当前是 tab 页时，点击 tab 时触发 （mpvue 0.0.16 支持）

**注意点 ：**

created：这个钩子是有bug的，所有页面的created函数会在项目加载的时候被一起调用，而且进入页面的时候不会在被调用，所以这个函数基本就不能使用了，用小程序的onLoad钩子代替

mounted：你如果从页面B返回页面A，页面A的mounted钩子不会被触发，因为页面没有被重新加载，如果有需要每次页面展示都要调用的逻辑，使用小程序的onShow代替

**2、mpvue转换的部分规则**

1. 微信小程序的页面的 query 参数是通过 onLoad 获取的，mpvue 对此进行了优化，直接通过 this.$root.$mp.query 获取相应的参数数据
2. 小程序里所有的 BOM／DOM 都不能用，因此v-html 、 v-text 不能用。

el：this.$refs...styles.width=offsetWIdth --> :style="{'width':offsetWidth}"

* 1

获取节点信息，

wx.createSelectorQuery().select(className).boundingClientRect().exec(res=>{

在此处获取到节点的信息： left,top,width,height})

* 1
* 2

1. 不支持部分复杂的 JavaScript 渲染表达式，我们会把 template 中的 {{}} 双花括号的部分，直接编码到 wxml 文件中，由于微信小程序的能力限制(数据绑定)，所以无法支持复杂的 JavaScript 表达式。

目前可以使用的有 + - \* % ?: ! == === > < [] .，剩下的还待完善。

{{ message.split('').reverse().join('') }}

4） 不支持过滤器，渲染部分会转成 wxml ，wxml 不支持过滤器，所以这部分功能不支持。

5） 不支持在 template 内使用 methods 中的函数。

6） 不支持 官方文档： Class 与 Style 绑定 中的 classObject 和 styleObject 语法。

<template>

<!-- 支持 -->

<div class="container" :class="computedClassStr"></div>

<div class="container" :class="{active: isActive}"></div>

<!-- 不支持 -->

<div class="container" :class="computedClassObject"></div>

</template>

<script>

export default {

data () {

return {

isActive: true

}

},

computed: {

computedClassStr () {

return this.isActive ? 'active' : ''

},

computedClassObject () {

return { active: this.isActive }

}

}

}

</script>

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24

7） 不支持在组件上使用 Class 与 Style 绑定，将class与style绑定在组件最外层div上

8） 列表渲染 需要注意一点， 嵌套列表渲染，必须指定不同的索引！

<template>

<ul v-for="(card, index) in list">

<li v-for="(item, itemIndex) in card">

{{item.value}}

</li>

</ul>

</template>

* 1
* 2
* 3
* 4
* 5
* 6
* 7

1. 小程序不支持路由，因此，路由跳转使用小程序的页面导航 api代替

this.$router.push-->wx.navigateTo() //进入子页面

this.$router.replace-->wx.reLaunch()//打开新页面

* 1
* 2

10） 获取当前页面地址

this.$route.fullPath–>getCurrentPages()[0].route

11） 接口返回参数结构调整，小程序的 request请求接口返回的数据会在外层添加一个data

res:{

res:{ data:{

code:'000000', --> code:'000000',

data:{...} data:{...}

} }

}

* 1
* 2
* 3
* 4
* 5
* 6

12） 不支持本地图片用作背景图，可以使用网络图片、或者 base64，或者使用img、image标签

13） 上拉加载 /下拉刷新，选用小程序的全局api，scroll-view中无法使用

14） 不支持 css媒体查询，css样式避免标签选择器，不易于维护

15） mpvue-wxParse富文本解析

1）安装npm i mpvue-wxparse

2)组件内

<template>

<wxParse :content="article" />

</template>

<script>

import wxParse from 'mpvue-wxparse'

components: {wxParse},

</script>

<style>

@import url("~mpvue-wxparse/src/wxParse.css");

</style>