跨平台开发 uni-app

刘军　liujun

目录
content

1 扩展组件 uni-ui

2 跨端兼容实现

3 页面路由和传参

4 其它常用API

5 自定义组件

6 状态管理Pinia

7 uni-app项目实战

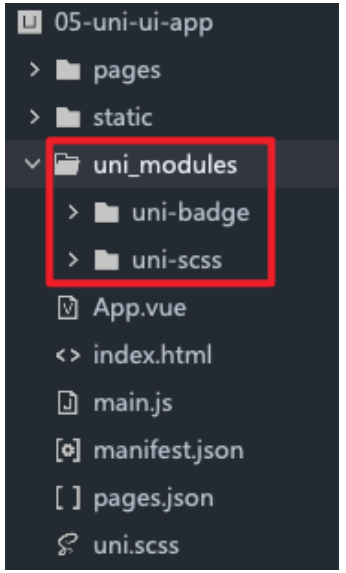
扩展组件（uni-ui）

- **什么是 uni-ui?**
  - uni-ui是DCloud提供的一个UI组件库，一套基于Vue组件、flex布局的跨全端UI框架。
  - uni-ui不包括uni-app框架提供的基础组件，而是基础组件的补充。
  - 详情：https://uniapp.dcloud.net.cn/component/uniui/uni-ui.html
- **uni-ui 特点**
  - 高性能
    - ✓ 目前为止，在小程序和混合app领域，uni-ui是性能的标杆。
    - ✓ 自动差量更新数据。uni-app引擎底层会自动用diff算法更新数据。
    - ✓ 优化逻辑层和视图层通讯折损。 比如，需要跟手式操作的UI组件，底层使用了wxs、bindingx等技术，实现了高性能的交互体验
      - ➢ WXS（WeiXin Script）是小程序的一套脚本语言，结合 WXML，可以构建出页面的结构。在 iOS 设备上小程序内的 WXS 会比 JavaScript 代码快 2 ~ 20 倍。
      - ➢ bindingx技术提供了一种称之为表达式绑定(Expression Binding) 的机制，在 weex 上让手势等复杂交互操作以60fps的帧率流畅执行，而不会导致卡顿。
  - 全端
    - ✓ uni-ui的组件都是多端自适应的，底层会抹平很多小程序平台的差异或bug。
    - ✓ uni-ui还支持nvue原生渲染、以及PC宽屏设备
  - 风格扩展
    - ✓ uni-ui的默认风格是中型的，与uni-app基础组件风格一致。
    - ✓ 支持uni.scss，可以方便的扩展和切换应用的风格。

安装 uni-ui 组件库

- **方式一（推荐）：通过 uni_modules（插件模块化规范）单独安装组件，**通过 uni_modules 按需安装某个组件：
  - ✓ 步骤1：官网找到扩展组件清单，然后将所需要的组件导入到项目，导入后直接使用，无需import和注册。
  - ✓ 步骤2：通常我们还想切换应用风格，这时可以在uni.scss导入uni-ui提供的内置scss变量，然后重启应用。
  - ✓ 注意：需要登录 DCloud 账号才能安装

组件名	组件说明
uni-badge	数字角标
uni-calendar	日历
uni-card	卡片
uni-collapse	折叠面板



- **方式二（推荐）：通过 uni_modules 导入全部组件**
  - 如想把所有uni-ui组件导入到项目，可以借用Hbuilder X插件导入。
  - 如没自动导入其他组件，可下载源码复制过去即可。
- **方式三：在 HBuilderX 新建 uni-app项目时，在模板中选择 uni-ui 模板来创建项目**
  - 由于uni-app独特的easycom（自动导包）技术，可以免引入、注册，就直接使用符合规则的vue组件。
- **方式四：npm安装**
  - 在 vue-cli 项目中可用 npm 安装 uni-ui 库

定制 uni-ui 主题风格

- 1.安装dart-sass插件(一般都会提示，并自动安装)
- 2.在项目根目录的uni.scss文件中引入uni-ui组件库的variable.scss变量文件，然后就可以使用或修改对应的scss变量。
  - @import '@/uni_modules/uni-scss/variables.scss';
- 3.变量主要定义的是主题色。

```scss
/* 需要放到文件最上面 */
@import '@/uni_modules/uni-scss/variables.scss';

/*
 以下变量是默认值，如不需要修改可以不用给下面的变量重新赋值
 */
// 主色
$uni-primary: #2979ff;
$uni-primary-disable:mix(#fff,$uni-primary,50%);
$uni-primary-light: mix(#fff,$uni-primary,80%);

// 辅助色
// 除了主色外的场景色，需要在不同的场景中使用（例如危险色表示危险的操作）
$uni-success: #18bc37;
$uni-success-disable:mix(#fff,$uni-success,50%);
$uni-success-light: mix(#fff,$uni-success,80%);
```

uni-forms 组件

- **uni-froms组件使用步骤（类似Element Plus的表单组件用法）：**
  - 安装uni-forms等组件
  - uni-forms搭建表单布局
  - 编写表单项的验证规则
  - 提交表单时验证表单项
  - 重置表单

rules 属性说明

每一个验证规则中，可以配置多个属性，下面是一些常见规则属性。实际上这里的规范，与uniCloud的DB Schema 规范相同。

属性名	类型	默认值	可选值	说明
required	Boolean	-	-	是否必填，配置此参数不会显示输入框左边的必填星号，如需要，请配置 `uni-forms-item` 组件的的required为true
range	Array	-	-	数组至少要有一个元素，且数组内的每一个元素都是唯一的。
format	String	-	-	内置校验规则，如这些规则无法满足需求，可以使用正则匹配或者自定义规则
pattern	RegExp	-	-	正则表达式，注意事项见下方说明
maximum	Number	-	-	校验最大值(大于)
minimum	Number	-	-	校验最小值(小于)
maxLength	Number	-	-	校验数据最大长度
errorMessage	String	-	-	校验失败提示信息语，可添加属性占位符，当前表格内属性都可用作占位符
validateFunction	Function	-	-	自定义校验规则

重写 uni-forms 组件样式

- 1.小程序、App直接重写，需要添加 important
- 2.H5、App和小程序使用：global(selector)，需要添加important
- 3.H5 、App和小程序使用：deep(selector)，需要添加important

```
// 重写样式（这个是微信小程序，  直接重写就行）
.uni-forms-item__label{
    justify-content: center !important;
    color:  red !important;
}


// 重写样式（h5支持 和 小程序 支持）
:global(.uni-forms-item__label) {
    color:  purple !important;
    justify-content: center !important;
}

// 重写样式( deep 深度样式，  h5支持）
:deep(.hy-form-item .uni-forms-item__label) {
    justify-content: center !important;
    color:  green !important;
}
```

跨平台兼容

- uni-app能实现一套代码、多端运行，核心是通过编译器 + 运行时实现的：
  - 编译器：将uni-app统一代码编译生成每个平台支持的特有代码；如在小程序平台，编译器将.vue文件拆分生成wxml、wxss、js等。
  - 运行时：动态处理数据绑定、事件代理，保证 Vue和对应宿主平台 数据的一致性；
- 跨平台存在的问题：
  - uni-app 已将常用的组件、JS API 封装到框架中，开发者按照 uni-app 规范开发即可保证多平台兼容，大部分业务均可直接满足。
  - 但每个平台有自己的一些特性，因此会存在一些无法跨平台的情况。
    - ✓ 大量写 if else，会造成代码执行性能低下和管理混乱。
    - ✓ 编译到不同的工程后二次修改，会让后续升级变的很麻烦。

条件编译写法	说明
#ifdef APP-PLUS 需条件编译的代码 #endif	仅出现在 App 平台下的代码
#ifndef H5 需条件编译的代码 #endif	除了 H5 平台，其它平台均存在的代码
#ifdef H5 || MP-WEIXIN 需条件编译的代码 #endif	在 H5 平台或微信小程序平台存在的代码（这里只有||，不可能出现&&，因为没有交集）

- 跨平台兼容解决方案：
  - 在 C 语言中，通过 #ifdef、#ifndef 的方式，为 windows、mac 等不同 os 编译不同的代码。
  - uni-app 参考这个思路，为 uni-app 提供了条件编译手段，在一个工程里优雅的完成了平台个性化实现。

条件编译

- 条件编译是用特殊的注释作为标记，在编译时根据这些特殊的注释，将注释里面的代码编译到不同平台。
- 具体的语法：以 #ifdef 或 #ifndef 加 **%PLATFORM%** 开头，以 #endif 结尾。
  - #ifdef：if defined 仅在某平台存在
  - #ifndef：if not defined 除了某平台，其它平台均存在
  - **%PLATFORM%**：平台名称
- 支持编写条件编译的文件，如下：
  - .vue（template 、script 、style）
  - .js 、.ts 、pages.json
  - .css、.scss、.less、.stylus
- 例如：设置页面的标题
  - H5专有API：document.title = ' '
  - 微信小程序专有API：wx.setNavigationBarTitle(object)

```
// #ifdef %PLATFORM%
平台特有的API实现
// #endif
```

%PLATFORM% 可取值如下：

值	生效条件
VUE3	HBuilderX 3.2.0+ 详情
APP-PLUS	App
APP-PLUS-NVUE或APP-NVUE	App nvue 页面
H5	H5
MP-WEIXIN	微信小程序
MP-ALIPAY	支付宝小程序
MP-BAIDU	百度小程序
MP-TOUTIAO	字节跳动小程序
MP-LARK	飞书小程序
MP-QQ	QQ小程序
MP-KUAISHOU	快手小程序
MP-JD	京东小程序
MP-360	360小程序
MP	微信小程序/支付宝小程序/百度小程序/字节跳动小程序/飞书小程序/QQ小程序/360小程序
QUICKAPP-WEBVIEW	快应用通用(包含联盟、华为)
QUICKAPP-WEBVIEW-UNION	快应用联盟
QUICKAPP-WEBVIEW-HUAWEI	快应用华为

注意事项

- 条件编译是利用注释实现的，在不同语法里注释写法不一样
  - js使用 // 注释
  - css 使用 /* 注释 */
  - vue/nvue 模板里使用 <!-- 注释 -->

```
// #ifdef %PLATFORM%
平台特有的API实现
// #endif
```

```
/* #ifdef %PLATFORM% */
平台特有样式
/* #endif */
```

```
<!-- #ifdef %PLATFORM% -->
平台特有的组件
<!-- #endif -->
```

- 条件编译 APP-PLUS 包含： APP-NVUE 和 APP-VUE
- APP-PLUS-NVUE 和 APP-NVUE 没什么区别，为了简写后面出了 APP-NVUE
- 使用条件编译请保证编译前和编译后文件的正确性，比如 json 文件中不能有多余的逗号
- Android 和 iOS 平台不支持条件编译，如需区分 Android、iOS 平台，请通过调用 uni.getSystemInfo 来获取平台信息
- 微信小程序主题色是绿色，而百度支付宝小程序是蓝色，应用想分平台适配颜色，条件编译是代码量最低、最容易维护的

新建Page页面

- uni-app页面是编写在pages目录下：
  - 可直接在 uni-app 项目上右键“新建页面”，HBuilderX会自动在pages.json中完成页面注册。
  - HBuilderX 还内置了常用的页面模板（如图文列表、商品列表等），这些模板可以大幅提升你的开发效率。
- 注意事项：
  - 每次新建页面，需在pages.json中配置pages列表（手动才需配置）
  - 未在pages.json -> pages 中配置的页面，uni-app会在编译阶段进行忽略。
- 删除页面：
  - 删除.vue文件 和 pages.json中对应的配置
- 配置tabBar
  - color
  - selectedColor
  - list -> pagePath、text、iconPath、selectedIconPath

页面路由

- uni-app 有两种页面路由跳转方式：使用navigator组件跳转、调用API跳转（类似小程序，与vue-router不同）。
  - 组件：navigator
  - API：navigateTo、redirectTo、navigateBack、switchTab

路由方式	页面栈表现	触发时机
初始化	新页面入栈	uni-app 打开的第一个页面
打开新页面	新页面入栈	调用 API uni.navigateTo 、使用组件 <navigator open-type="navigate"/>
页面重定向	当前页面出栈，新页面入栈	调用 API uni.redirectTo 、使用组件 <navigator open-type="redirectTo"/>
页面返回	页面不断出栈，直到目标返回页	调用 API uni.navigateBack 、使用组件 <navigator open-type="navigateBack"/> 、用户按左上角返回按钮、安卓用户点击物理back按键
Tab 切换	页面全部出栈，只留下新的 Tab 页面	调用 API uni.switchTab 、使用组件 <navigator open-type="switchTab"/> 、用户切换 Tab
重加载	页面全部出栈，只留下新的页面	调用 API uni.reLaunch 、使用组件 <navigator open-type="reLaunch"/>

```
<!-- 1.方式一 -->
<navigator url="/pages/home-detail/home-detail">
    <button type="default">跳转到home-detial页面</button>
</navigator>
```

```
goTolifeCycleVue3() {
    let params  = {
        name: 'liu jun ./ 刘军',
        age: 18
    }
    // let jsonStr = JSON.stringify(params) // ok
    let jsonStr = encodeURIComponent(JSON.stringify(params) )// ok
    uni.navigateTo({
        url:'/pages/lifecycle-vue3/lifecycle-vue3?info=' + jsonStr
    })
},
goToEventChannel() {
    let params  = {
```

页面间通讯

- **在uni-app中，常见页面通讯方式：**
  - 方式一：url查询字符串和EventChannel
  - 方式二：使用事件总线
  - 方式三：全局数据 globalData
  - 方式四：本地数据存储
  - 方式五：Vuex和Pinia，状态管理库。
- **方式一：url和EventChannel(兼容h5、weapp、app)**
  - 直接在url后面通过查询字符串的方式拼接
    - ✓ 如url查询字符串出现特殊字符等格式，需编码
  - 然后可在onLoad生命周期中获取url传递的参数
- EventChannel 对象的获取方式
  - Options语法：this.getOpenerEventChannel()
  - Composition语法：getCurrentInstance().proxy. getOpenerEventChannel()

url有长度限制，太长的字符串会传递失败，可改用窗体通信、全局变量，另外参数中出现空格等特殊字符时需要对参数进行编码，如下为使用 `encodeURIComponent` 对参数进行编码的示例。

```html
<navigator :url="'/pages/test/test?item='+ encodeURIComponent(JSON.stringify(item))"></navigator>
```

```js
// 在test.vue页面接受参数
onLoad: function (option) {
    const item = JSON.parse(decodeURIComponent(option.item));
}
```

```vue
// vue3 全平台新增：通过 props 来获取页面参数的使用方式
const props = defineProps({
    name: String,
    age: String,
});

console.log('props=>', props.name);

onLoad((option)=>{
    // 接收页面传递参数 (vue3 全平台新增：通过 props 来获取页面参数的使用方式)
    console.log('lifecycle onLoad', option);
})
```

事件总线

- **方式二：事件总线**
  - **uni.$emit(eventName, OBJECT)** 触发全局的自定义事件。
  - **uni.$on(eventName, callback)** 监听全局的自定义事件。由 uni.$emit 触发。
  - **uni.$once(eventName, callback)** 只监听一次全局的自定义事件。由 uni.$emit 触发
  - **uni.$off(eventName, callback)** 移除全局自定义事件监听器。
    - ✓ 如果没有提供参数，则移除所有的事件监听器；

- **注意事项：**
  - 需先监听，再触发事件，比如：你在A界面触发，然后跳转到B页面后才监听是不行的。
  - 通常on 和 off 是同时使用，可以避免多次重复监听
  - 适合页面返回传递参数、适合跨组件通讯，不适合界面跳转传递参数

页面生命周期(Options API)

- uni-app 常用的页面生命周期函数：
  - onLoad(options) -> onLoad
  - onShow -> onShow
  - onReady -> onReady
  - onHide -> onHide
  - onUnload -> onUnload
  - onPullDownRefresh -> onPullDownRefresh
  - onReachBottom -> onReachBottom
  - 更多： https://uniapp.dcloud.net.cn/tutorial/page.html#lifecycle

函数名	说明	平台差异说明	最低版本
onInit	监听页面初始化，其参数同 onLoad 参数，为上个页面传递的数据，参数类型为 Object（用于页面传参），触发时机早于 onLoad	百度小程序	3.1.0+
onLoad	监听页面加载，其参数为上个页面传递的数据，参数类型为 Object（用于页面传参），参考示例		
onShow	监听页面显示。页面每次出现在屏幕上都触发，包括从下级页面点返回露出当前页面		
onReady	监听页面初次渲染完成。注意如果渲染速度快，会在页面进入动画完成前触发		
onHide	监听页面隐藏		
onUnload	监听页面卸载		
onResize	监听窗口尺寸变化	App、微信小程序、快手小程序	
onPullDownRefresh	监听用户下拉动作，一般用于下拉刷新，参考示例		
onReachBottom	页面滚动到底部的事件（不是scroll-view滚到底），常用于下拉下一页数据。具体见下方注意事项		
onTabItemTap	点击 tab 时触发，参数为Object，具体见下方注意事项	微信小程序、QQ小程序、支付宝小程序、百度小程序、H5、App、快手小程序、京东小程序	
onShareAppMessage	用户点击右上角分享	微信小程序、QQ小程序、支付宝小程序、字节小程序、飞书小程	

- 注意事项：
  - 页面可以使用Vue组件生命周期吗？ 可以的
  - 页面滚动才会触发 onReachBottom 回调，如果自行通过overflow实现的滚动不会触发 onReachBottom 回调

页面生命周期(Composition API)

- uni-app 常用的页面生命周期函数:
  - onLoad -> onLoad
  - onShow -> onShow
  - onReady -> onReady
  - onHide -> onHide
  - onUnload -> onUnload
  - onResize -> onResize
  - onPullDownRefresh -> onPullDownRefresh
  - onReachBottom -> onReachBottom
  - 更多: https://uniapp.dcloud.net.cn/tutorial/page.html#lifecycle

函数名	说明	平台差异说明	最低版本
onInit	监听页面初始化，其参数同 onLoad 参数，为上个页面传递的数据，参数类型为 Object（用于页面传参），触发时机早于 onLoad	百度小程序	3.1.0+
onLoad	监听页面加载，其参数为上个页面传递的数据，参数类型为 Object（用于页面传参），参考示例		
onShow	监听页面显示。页面每次出现在屏幕上都触发，包括从下级页面点返回露出当前页面		
onReady	监听页面初次渲染完成。注意如果渲染速度快，会在页面进入动画完成前触发		
onHide	监听页面隐藏		
onUnload	监听页面卸载		
onResize	监听窗口尺寸变化	App、微信小程序、快手小程序	
onPullDownRefresh	监听用户下拉动作，一般用于下拉刷新，参考示例		
onReachBottom	页面滚动到底部的事件（不是scroll-view滚到底），常用于下拉下一页数据。具体见下方注意事项		
onTabItemTap	点击 tab 时触发，参数为Object，具体见下方注意事项	微信小程序、QQ小程序、支付宝小程序、百度小程序、H5、App、快手小程序、京东小程序	
onShareAppMessage	用户点击右上角分享	微信小程序、QQ小程序、支付宝小程序、字节小程序、飞书小程	

网络请求

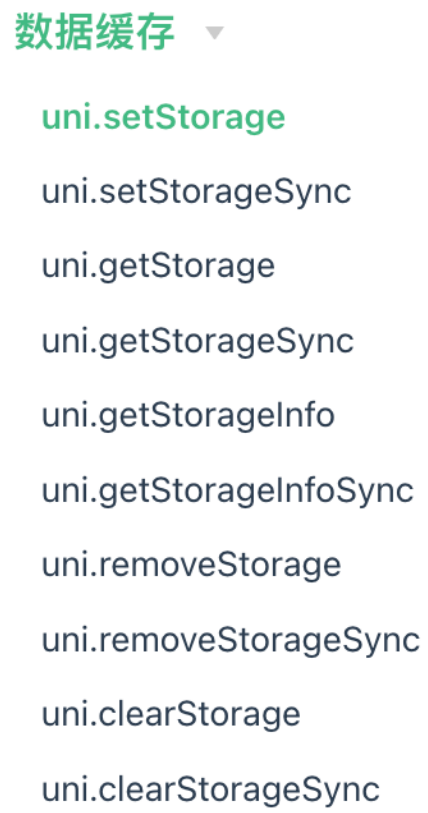
- **uni.request(OBJECT)** 发起网络请求。
  - 登录各个小程序管理后台，给网络相关的 API 配置合法域名（域名白名单）
  - 微信小程序开发工具，在开发阶段可以配置：不校验合法域名
  - 运行到手机时，资源没有出来时可以打开手机的调试模式
  - 请求的 header 中 content-type 默认为 application/json

参数名	类型	必填	默认值	说明	平台差异说明
url	String	是		开发者服务器接口地址	
data	Object/String/ArrayBuffer	否		请求的参数	App 3.3.7 以下不支持 ArrayBuffer 类型
header	Object	否		设置请求的 header，header 中不能设置 Referer。	App、H5端会自动带上cookie，且H5端不可手动修改
method	String	否	GET	有效值详见下方说明	
timeout	Number	否	60000	超时时间，单位 ms	H5(HBuilderX 2.9.9+)、APP(HBuilderX 2.9.9+)、微信小程序（2.10.0）、支付宝小程序

method	App	H5	微信小程序	支付宝小程序	百度小程序	字节跳动小程序、飞书小程序	快手小程序	京东小程序
GET	√	√	√	√	√	√	√	√
POST	√	√	√	√	√	√	√	√
PUT	√	√	√	x	√	√	x	x
DELETE	√	√	√	x	√	x	x	x

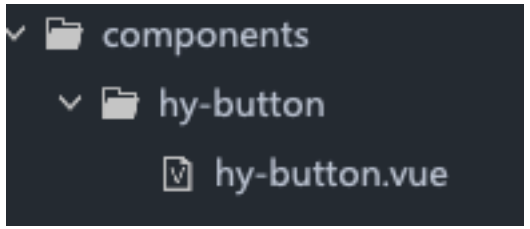
数据缓存

- **uni.setStorage(OBJECT)**
  - 将数据存储在本地缓存中指定的 key 中，会覆盖掉原来该 key 对应的内容，这是一个异步接口。
- **uni.setStorageSync(KEY, DATA)**
  - 将 data 存储在本地缓存中指定的 key 中，会覆盖掉原来该 key 对应的内容，这是一个同步接口。
- **uni.getStorage(OBJECT)**
  - 从本地缓存中异步获取指定 key 对应的内容。
- **uni.getStorageSync(KEY)**
  - 从本地缓存中同步获取指定 key 对应的内容。
- **uni.removeStorage(OBJECT)**
  - 从本地缓存中异步移除指定 key。
- **uni.removeStorageSync(KEY)**
  - 从本地缓存中同步移除指定 key。



组件（Component）

- uni-app 组件 Vue标准组件基本相同，但是也有一点区别，比如：
  - 传统vue组件，需要创建组件、引用、注册，三个步骤后才能使用组件， easycom组件模式可以将其精简为一步。
  - easycom组件规范：
    - ✓ 组件需符合components/组件名称/组件名称.vue 的目录结构。
    - ✓ 符合以上目录结构的就可不用引用、注册，直接在页面中使用该组件了。



组件生命周期

- uni-app 组件支持的生命周期，与Vue组件的生命周期相同。
  - 组件中可以使用页面的生命周期吗?
    - ✓ 在Options API 语法：组件中不支持使用页面生命周期。
    - ✓ 在Composition API语法：组件中支持页面生命周期，不同端支持情况有差异。

函数名	说明	平台差异说明	最低版本
beforeCreate	在实例初始化之前被调用。详见		
created	在实例创建完成后被立即调用。详见		
beforeMount	在挂载开始之前被调用。详见		
mounted	挂载到实例上去之后调用。详见 注意：此处并不能确定子组件被全部挂载，如果需要子组件完全挂载之后在执行操作可以使用 `$nextTick` Vue官方文档		
beforeUpdate	数据更新时调用，发生在虚拟 DOM 打补丁之前。详见	仅H5平台支持	
updated	由于数据更改导致的虚拟 DOM 重新渲染和打补丁，在这之后会调用该钩子。详见	仅H5平台支持	
beforeDestroy	实例销毁之前调用。在这一步，实例仍然完全可用。详见		
destroyed	Vue 实例销毁后调用。调用后，Vue 实例指示的所有东西都会解绑定，所有的事件监听器会被移除，所有的子实例也会被销毁。详见		

- `beforeCreate` → use `setup()`
- `created` → use `setup()`
- `beforeMount` → `onBeforeMount`
- `mounted` → `onMounted`
- `beforeUpdate` → `onBeforeUpdate`
- `updated` → `onUpdated`
- `beforeDestroy` → `onBeforeUnmount`
- `destroyed` → `onUnmounted`
- `errorCaptured` → `onErrorCaptured`

Vue3 Options API

Options: State

- data
- props
- computed
- methods
- watch
- emits
- expose

Options: Rendering

- template
- render
- compilerOptions

Options: Lifecycle

- beforeCreate
- created
- beforeMount
- mounted
- beforeUpdate
- updated
- beforeUnmount
- unmounted
- errorCaptured
- renderTracked
- renderTriggered
- activated
- deactivated
- serverPrefetch

Options: Composition

- provide
- inject
- mixins
- extends

Options: Misc

- name
- inheritAttrs
- components
- directives

Component Instance

- $data
- $props
- $el
- $options
- $parent
- $root
- $slots
- $refs
- $attrs
- $watch()
- $emit()
- $forceUpdate()
- $nextTick()

Vue3 Composition API

setup()

- Basic Usage
- Accessing Props
- Setup Context
- Usage with Render Functions

Reactivity: Core

- ref()
- computed()
- reactive()
- readonly()
- watchEffect()
- watchPostEffect()
- watchSyncEffect()
- watch()

Reactivity: Utilities

- isRef()
- unref()
- toRef()
- toRefs()
- isProxy()
- isReactive()
- isReadonly()

Reactivity: Advanced

- shallowRef()
- triggerRef()
- customRef()
- shallowReactive()
- shallowReadonly()
- toRaw()
- markRaw()
- effectScope()
- getCurrentScope()
- onScopeDispose()

Lifecycle Hooks

- onMounted()
- onUpdated()
- onUnmounted()
- onBeforeMount()
- onBeforeUpdate()
- onBeforeUnmount()
- onErrorCaptured()
- onRenderTracked()
- onRenderTriggered()
- onActivated()
- onDeactivated()
- onServerPrefetch()

Dependency Injection

- provide()
- inject()

Pinia

- 认识Pinia
  - Pinia（发音为 /piːnjʌ/，如英语中的 peenya） 是 Vue 的存储库，它允许跨组件、页面共享状态。
  - uni-app 内置了 Pinia，使用 HBuilder X 不需要手动安装，直接使用即可。
  - 使用 CLI 需要手动安装，执行 yarn add pinia 或 npm install pinia。

- Pinia的初体验，步骤如下：
  - 第一步：在 main.js 中安装 Pinia插件
    - ✓ app.use(Pinia.createPinia());
  - 第二步：接着创建一个store
  - 第三步：然后在组件中就可以直接使用了

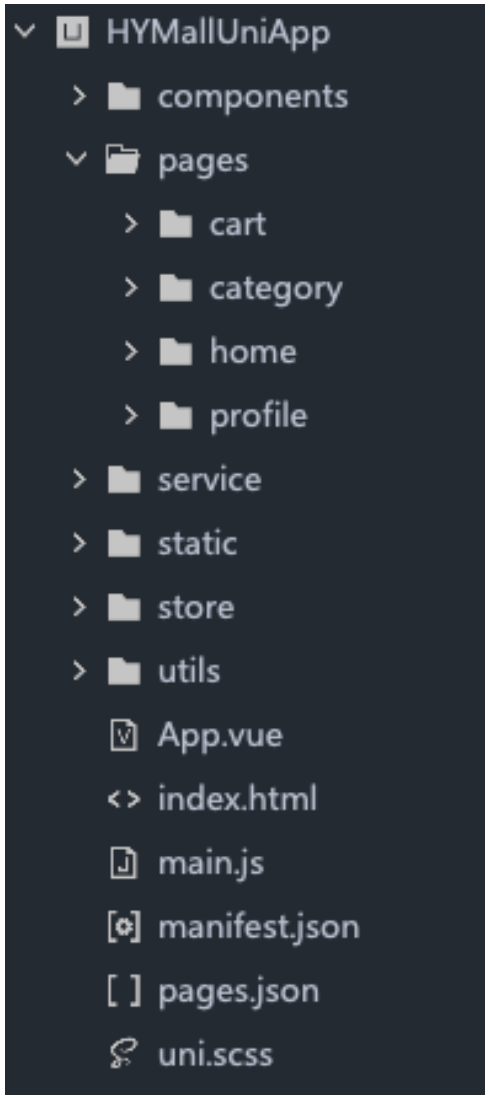
```
import * as Pinia from 'pinia';

export function createApp() {
    const app = createSSRApp(App);
    app.use(Pinia.createPinia());
    return {
        app,
        Pinia, // 此处必须将 Pinia 返回
    };
}
```

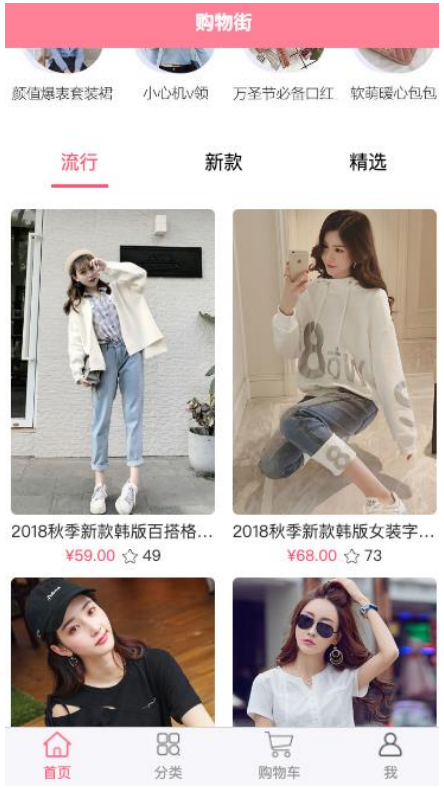
```
// stores/counter.js
import { defineStore } from 'pinia';

export const useCounterStore = defineStore('counter', {
    state: () => {
        return { count: 0 };
    },
    // 也可以这样定义
    // state: () => ({ count: 0 })
    actions: {
        increment() {
            this.count++;
        },
    },
});
```

项目目录结构


HYMallUniApp
components
pages
cart
category
home
profile
service
static
store
utils
App.vue
index.html
main.js
manifest.json
pages.json
uni.scss


购物街
美丽福利购
焕新美颜
十点抢券
好物特卖
内购福利
初秋上新
本周流行
霸街长袖裙
时尚懒人套装
微胖遮肉穿搭
降温保暖鞋
颜值爆表套装裙
小心机v领
万圣节必备口红
软萌暖心包包
首页
分类
购物车
我


购物街
颜值爆表套装裙
小心机v领
万圣节必备口红
软萌暖心包包
流行
新款
精选
2018秋季新款韩版百搭格...
¥59.00 49
2018秋季新款韩版女装字...
¥68.00 73
首页
分类
购物车
我

微信小程序-打包配置

- 1.注册一个小程序账号：_https://mp.weixin.qq.com/wxopen/waregister?action=step1
- 2. 登录已注册好的账号，拿到小程序 APPID： wxbc30134b589795b0（需要用你自己申请的）
- 3.修改一下manifest.json的配置（比如：appid、es6-es5、压缩）
- 4.发行->打包微信小程序
- 5.在微信开发者工具中点击 上传 代码

H5-打包配置（一）


基础配置
App图标配置
App启动界面配置
App模块配置
App权限配置
App原生插件配置
App常用其它设置
Web配置
微信小程序配置
百度小程序配置
字节跳动小程序配置
支付宝小程序配置
QQ小程序配置
快手小程序配置
Web配置
配置指南
页面标题
默认为应用名称
美丽说
页面的标题，title标签
index.html模板路径
默认为空，可定制生成的html代码，自定义meta、引入外部js，参考
路由模式
应用路由模式
hash|history，二选一
hash
运行的基础路径
资源的前缀，即publicPath
例：/h5/，代表在域名的/h5目录下部署运行。如设为 ./，则代表相对路
启用https协议
配置 开发环境 devServer 支持协议
前端开发服务端口
详情
8888
配置开发环境 devServer 的端口
发行时启用摇树优化（自动裁剪没有使用的组件和API库）
可减少网站体积和加快首页渲染速度。参考文档


H5发行
cli程序化部署教程
网站标题 美丽说
网站域名 输入网站域名
生成sourcemap（可用于uni统计的错误分析） 详情
以SSR方式发行 帮助
将编译后的资源部署到前端网页托管 详情
欢迎开通 uniAD 广告进行变现，申请入口 | 开发文档
取消
发行

H5-打包配置（二）

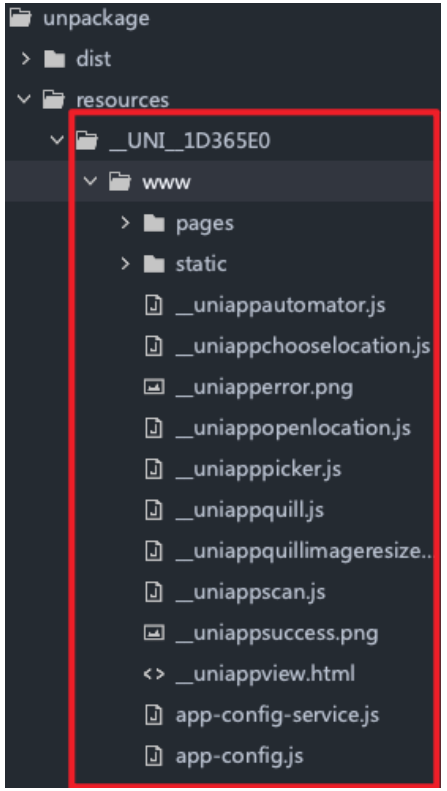
- 1.购买阿里云服务器
- 2.连接阿里云服务器（VSCode 安装 Remove SSH 插件）
- 3.安装Nginx服务器
  - sudo yum install nginx # 安装 nginx
  - sudo systemctl enable nginx # 设置开机启动
- 4.启动Nginx服务器(　http://8.134.149.197　)
  - sudo service nginx start # 启动 nginx 服务
  - sudo service nginx restart # 重启 nginx 服务
- 5.修改Nginx的配置(/etc/nginx/nginx.conf)
  - 切换为 root 用户，修改部署路径
- 6.打包和部署项目

Android-云打包配置

- 1.注册一个Dcloud账号： https://dev.dcloud.net.cn/ 或在 HBuilder X 中注册
- 2. HBuilder X 登录已注册好的账号，然后在manifest.json中配置应用基本信息
- 3.云打包Android时，会自动生成证书（也可以手动生成）
- 4.开始执行云打包

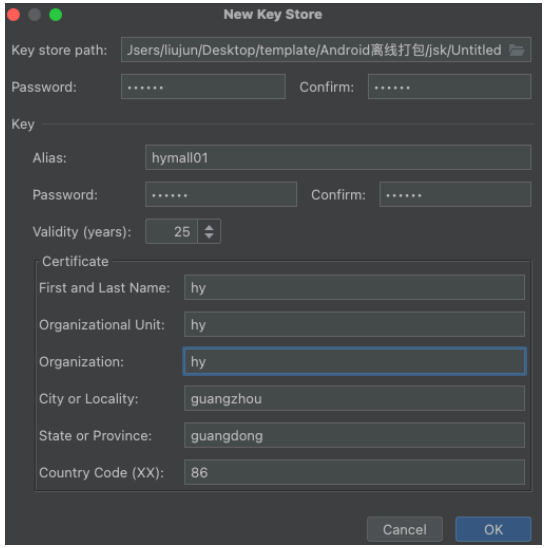
Android-离线本地打包(一)

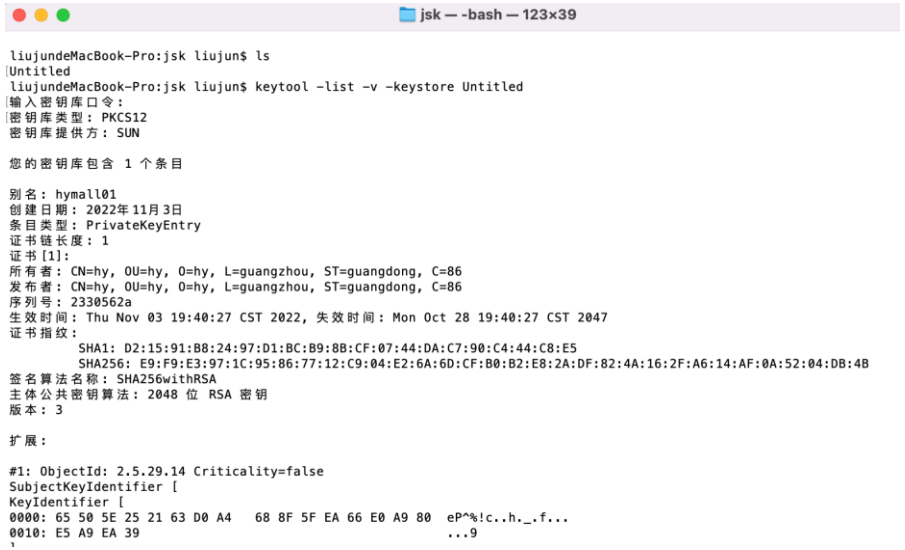
- 生成本地打包App资源
  - 先完成社区身份验证，请点击链接 https://ask.dcloud.net.cn/account/setting/profile
  - 验证后再重新打包：原生App-本地打包->生成本地打包App资源



Android-离线本地打包（二）

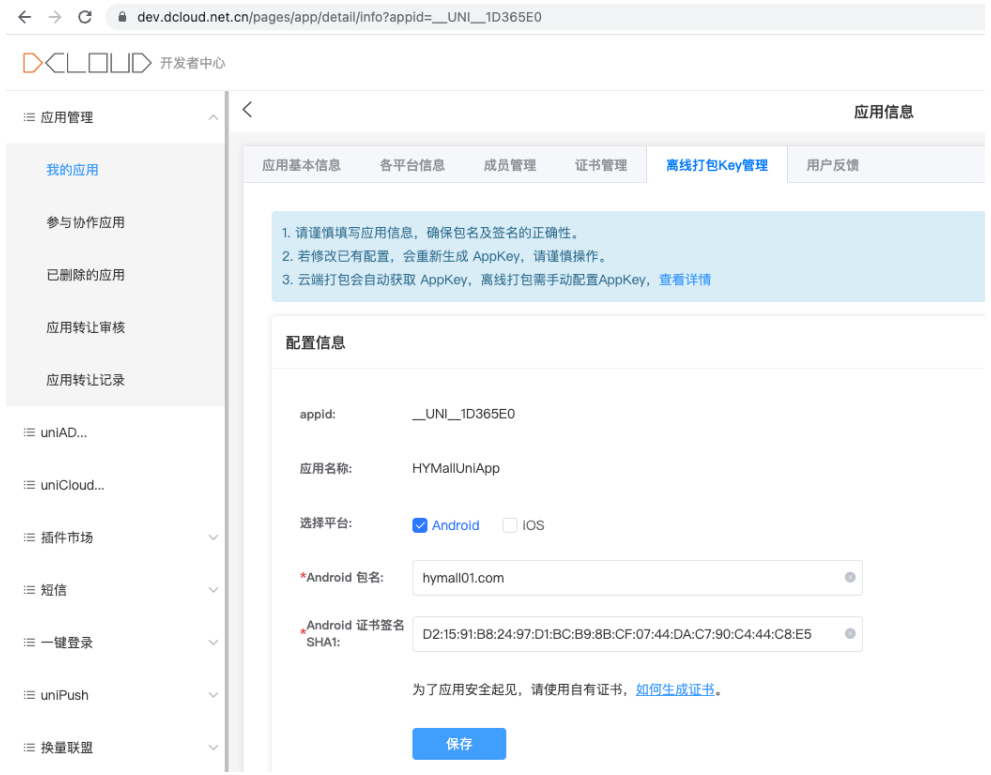
- 生成证书(jks文件)
  - 文件密码
  - 证书别名：hymall01
  - 证书密码
- 通过证书获取sha1 （https://ask.dcloud.net.cn/article/35777）
  - keytool -list -v -keystore 生成的证书文件
    - ✓ D2:15:91:B8:24:97:D1:BC:B9:8B:CF:07:44:DA:C7:90:C4:44:C8:E5





Android-离线本地打包（三）

- 获取Appkey
  - 登录DCloud开发者中心：https://dev.dcloud.net.cn/pages/app/list
  - 我的应用中找到需要打包的应用，然后点击该应用查看应用的信息，选择离线打包key管理
    - ✓ Appkey：58129cb2c328630a7416913c58ba6a9f



Android-离线本地打包（四）

- 离线打包apk常见配置（项目路径不支持中文）：
  - 配置Appkey: Appkey
  - 配置应用版本号: versionCode
  - 应用的版本名称: versionName
  - 应用的包名，一般设置为反向域名： applicationId
  - SDK最低支持版本21： minSdkVersion
  - 配置应用名称app_name ， 建议与manifest.json中name同为一个
  - 图标名称：
    - ✓ icon.png为应用的图标。
    - ✓ splash.png为应用启动页的图标。
    - ✓ push.png为推送消息的图标。
  - 资源配置，包括：导出的app资源和data资源
  - dcloud_control.xml中的appid为拷贝过来的uni-app的id