uni-app 微信小程序项目实战

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

第一天

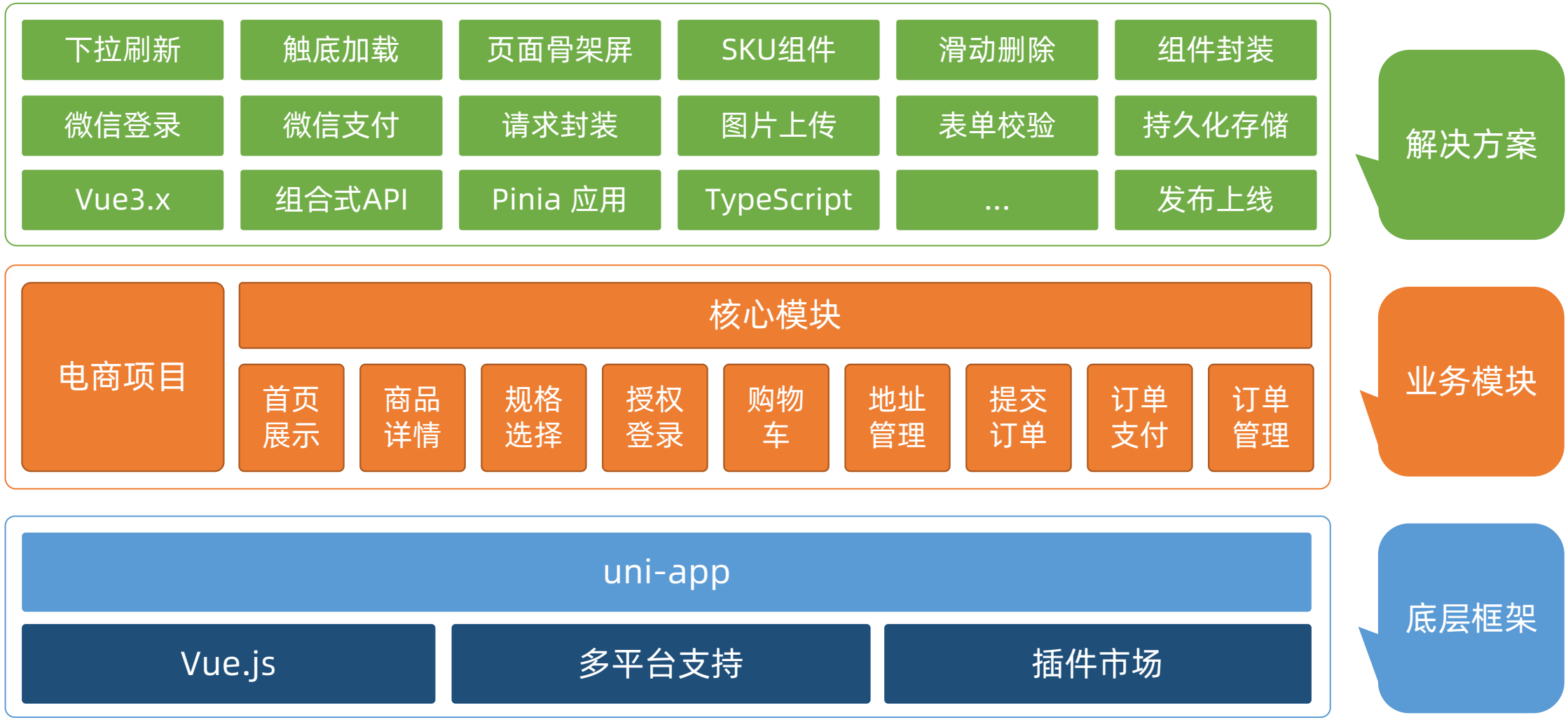
uni-app基础 + 项目起步

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

项目架构

uni-app 小兔鲜儿电商项目架构



小兔鲜儿电商课程安排


uni-app基础 + 项目起步
（day01）
登录模块 + 地址模块
（day04）
首页模块
（day02）
SKU模块 + 购物车模块
（day05）
推荐 + 分类 + 详情
（day03）
订单模块 + 打包上线
（day06）

创建 uni-app 项目

创建 uni-app 项目

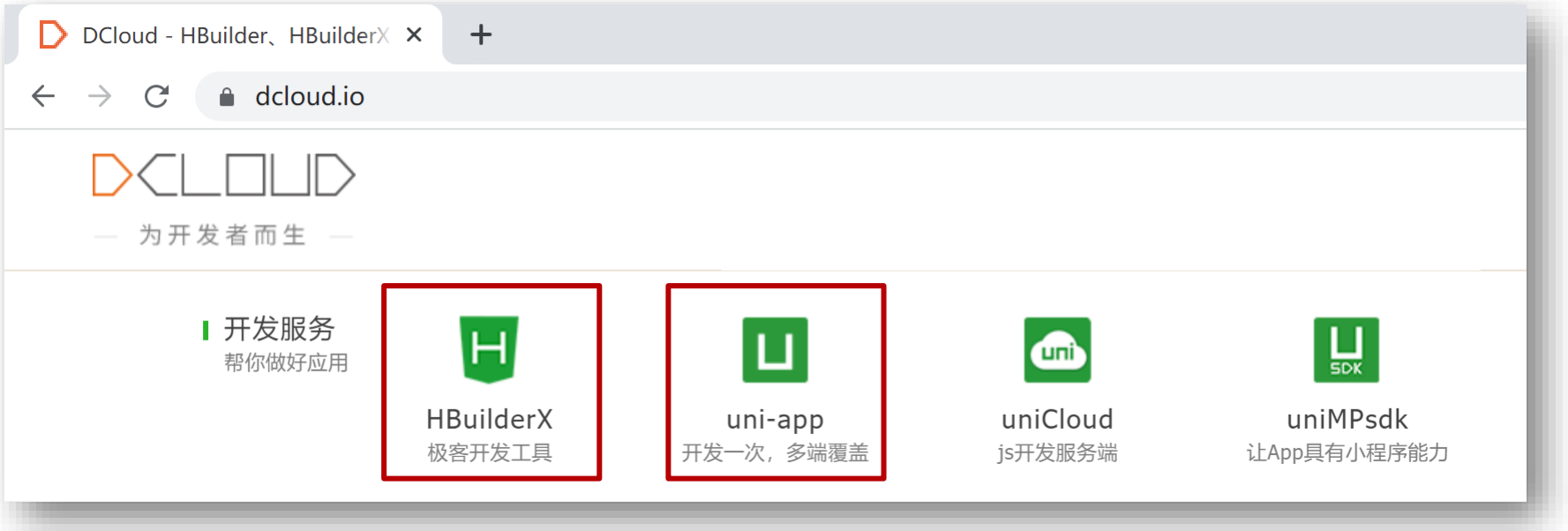
uni-app 支持两种方式创建项目：

1. 通过 HBuilderX 创建
2. 通过命令行创建

创建 uni-app 项目

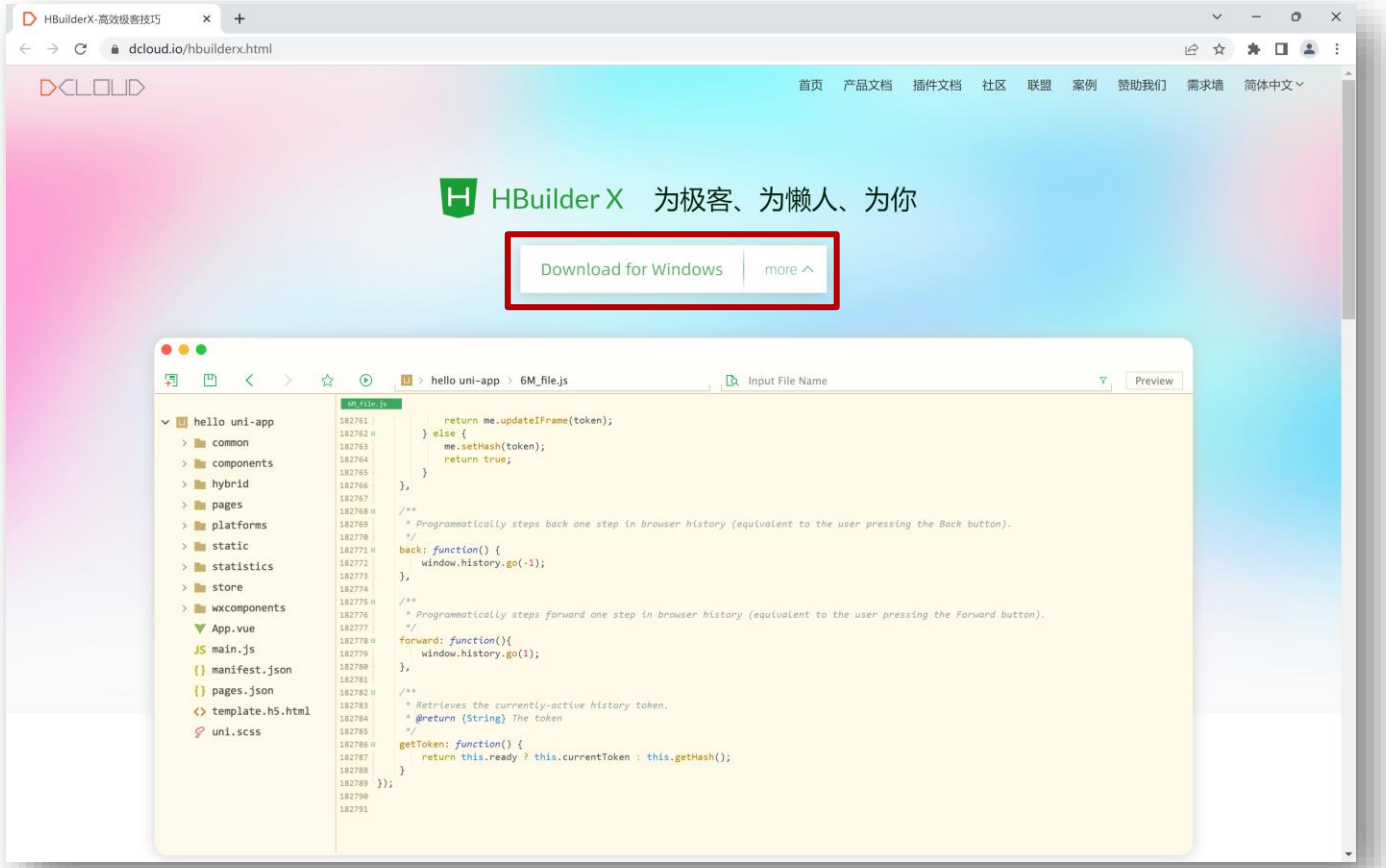
uni-app 支持两种方式创建项目：

1. 通过 HBuilderX 创建
2. 通过命令行创建



通过 HBuilderX 创建 uni-app 项目

1.1 下载安装 HbuilderX 编辑器



通过 HBuilderX 创建 uni-app 项目

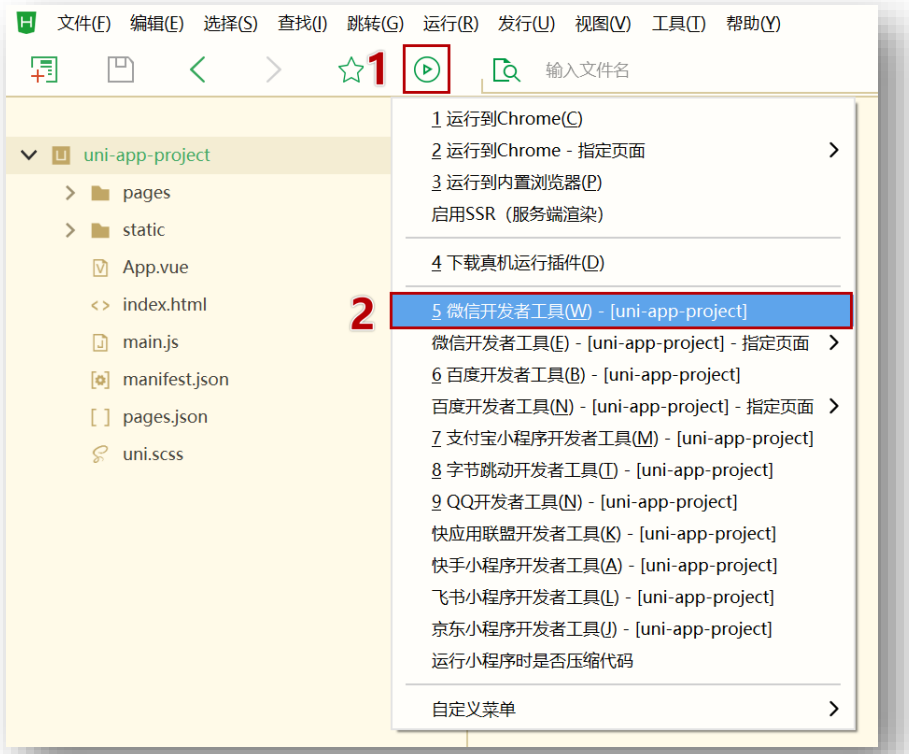
1.2 通过 HbuilderX 创建 uni-app vue3 项目

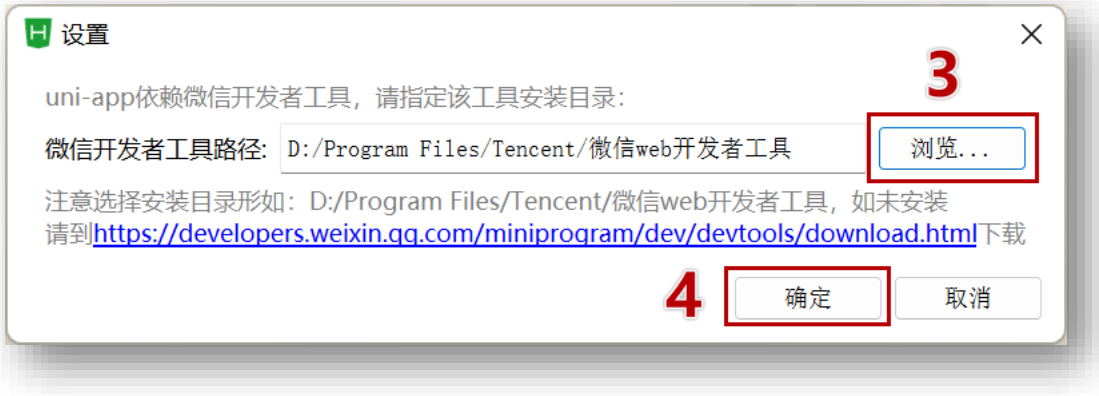
通过 HBuilderX 创建 uni-app 项目

1.3 安装 uni-app vue3 编译器插件

通过 HBuilderX 创建 uni-app 项目

1.4 编译成微信小程序端代码

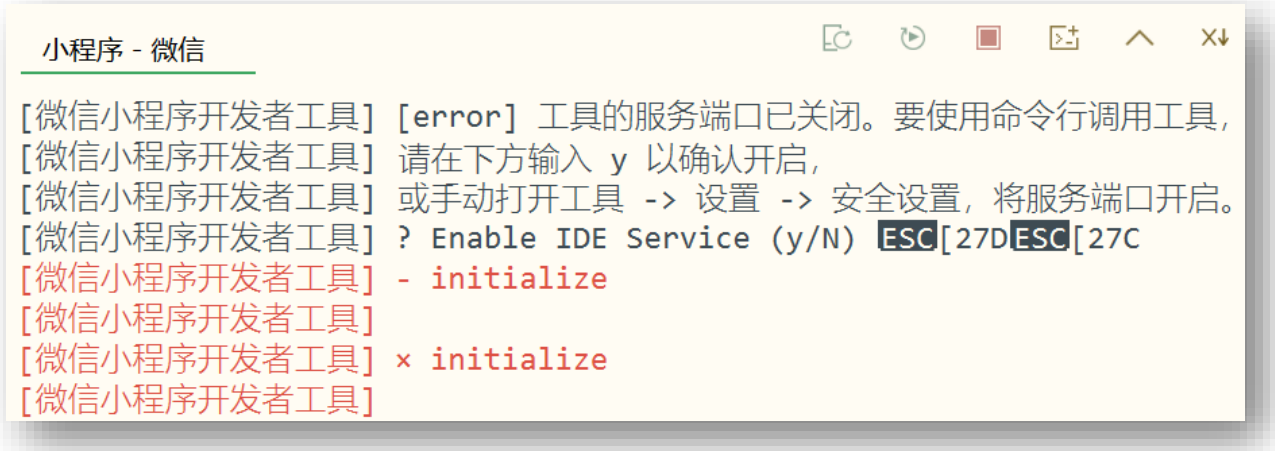




温馨提示：路径只需初次使用时设置一次即可

通过 HBuilderX 创建 uni-app 项目

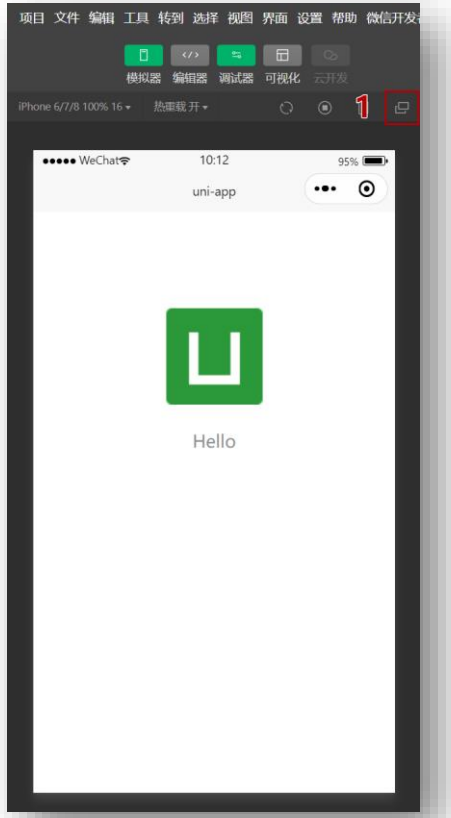
1.5 开启服务端口

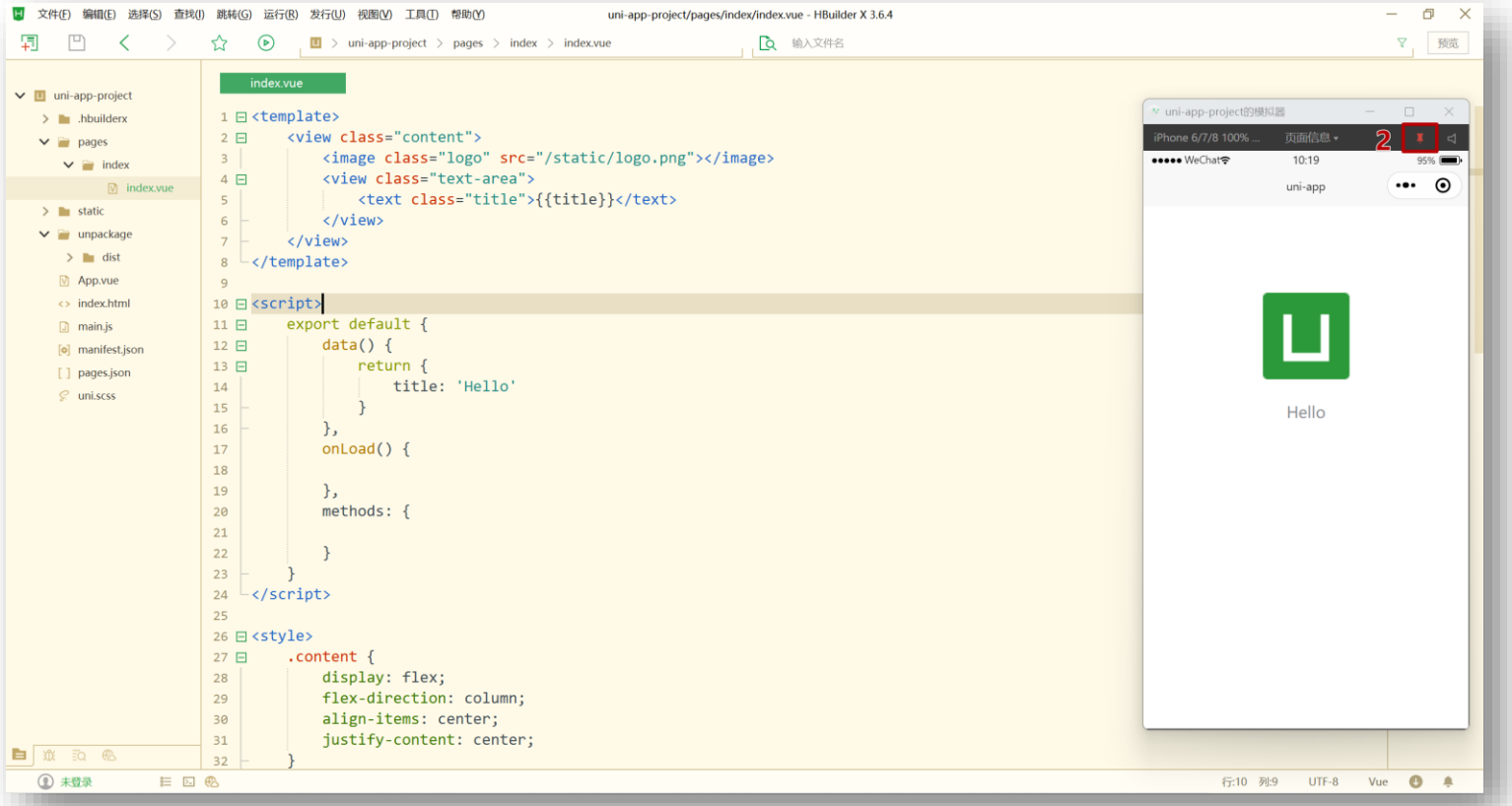


温馨提示：服务端口只需初次使用时开启一次即可

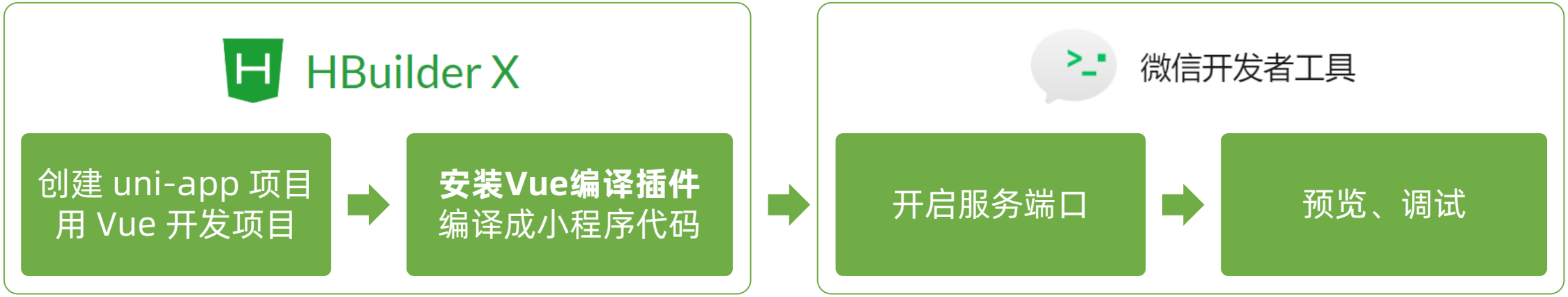
通过 HBuilderX 创建 uni-app 项目

小技巧分享：模拟器窗口分离和置顶





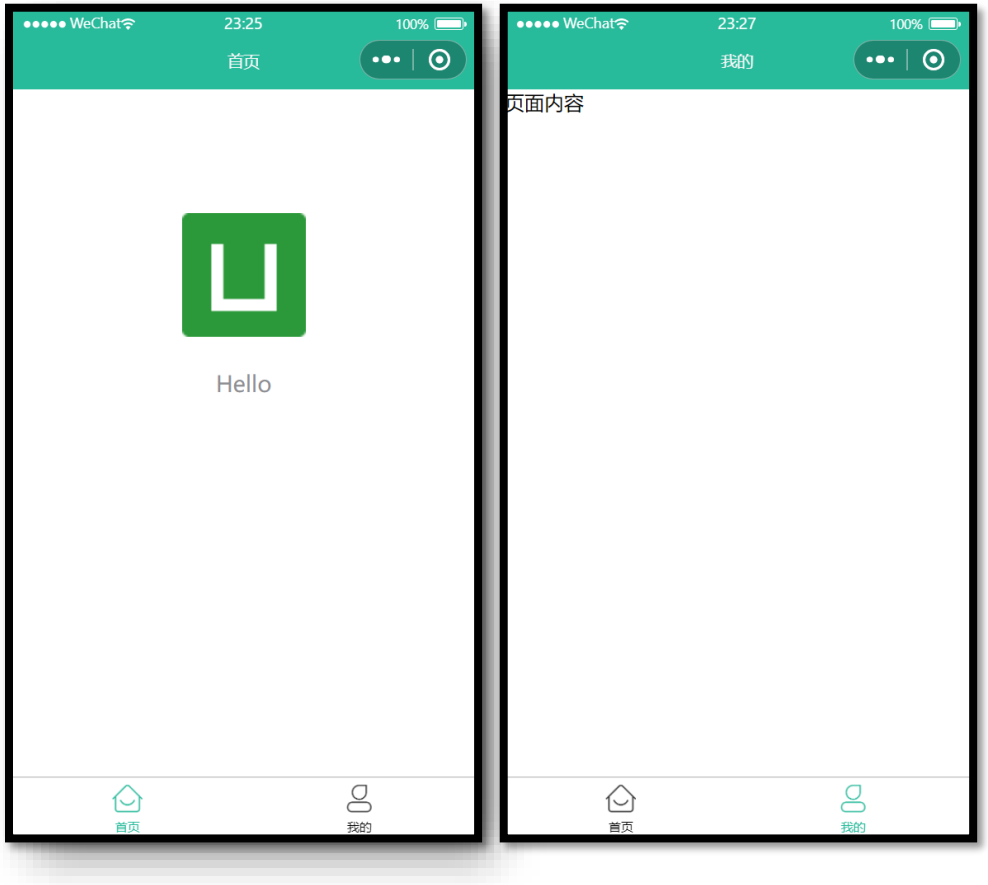
通过 HBuilderX 创建 uni-app 项目



pages.json 和 tabBar 案例

pages.json 和 tabBar 案例

```
├─pages              业务页面文件存放的目录
│  └─index
│      └─index.vue   index页面
├─static             存放应用引用的本地静态资源的目录(注意：静态资源只能存放于此)
├─unpackage          非工程代码，一般存放运行或发行的编译结果
├─index.html         H5端页面
├─main.js            Vue初始化入口文件
├─App.vue            配置App全局样式、监听应用生命周期
├─pages.json         配置页面路由、导航栏、tabBar等页面类信息
├─manifest.json      配置appid、应用名称、logo、版本等打包信息
└─uni.scss           uni-app内置的常用样式变量
```



pages.json 和 tabBar 案例

pages.json

- pages
  页面路由及窗口表现

```
"pages": [
    {
        "path": "pages/index/index",
        "style": {
            "navigationBarTitleText": "首页"
        }
    },
    {
        "path" : "pages/my/my",
        "style": {
            "navigationBarTitleText": "我的"
        }
    }
],
```

- globalstyle
  默认窗口表现

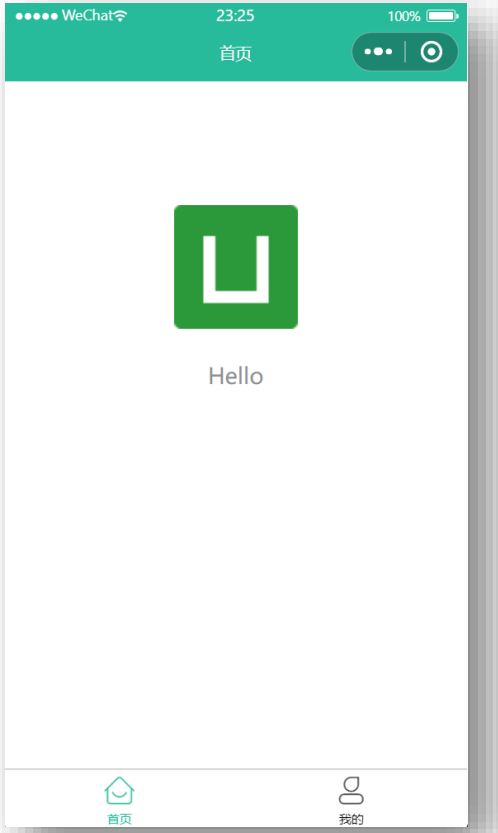
```
"globalStyle": {
    "navigationBarTextStyle": "white",
    "navigationBarTitleText": "uni-app",
    "navigationBarBackgroundColor": "#27BA9B",
    "backgroundColor": "#F8F8F8"
},
```

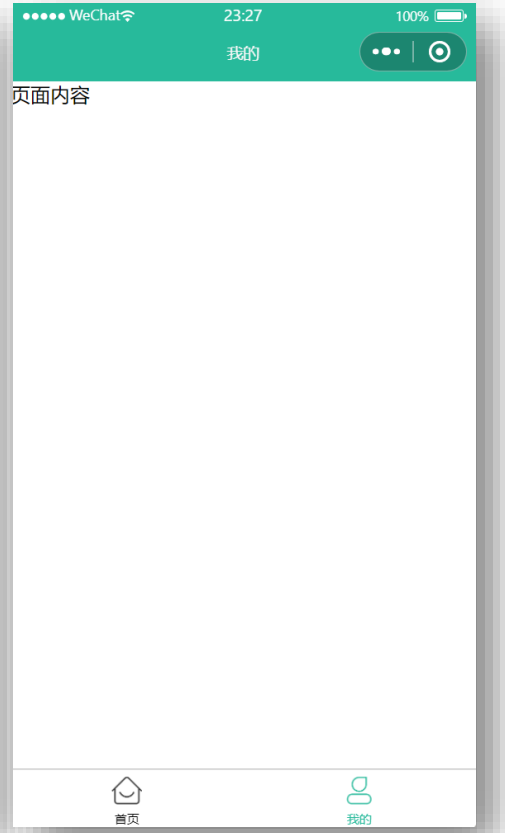
- tabBar
  至少两个才显示

```
"tabBar": {
    "selectedColor": "#27BA9B",
    "list": [
        {
            "pagePath": "pages/index/index",
            "text": "首页",
            "iconPath": "static/tabs/home_default.png",
            "selectedIconPath": "static/tabs/home_selected.png"
        },
        {
            "pagePath": "pages/my/my",
            "text": "我的",
            "iconPath": "static/tabs/user_default.png",
            "selectedIconPath": "static/tabs/user_selected.png"
        }
    ]
},
```

pages.json 和 tabBar 案例

```
├─pages              业务页面文件存放的目录
│  └─index
│      └─index.vue   index页面
├─static             存放应用引用的本地静态资源的目录(注意：静态资源只能存放于此)
├─unpackage          非工程代码，一般存放运行或发行的编译结果
├─index.html         H5端页面
├─main.js            Vue初始化入口文件
├─App.vue            配置App全局样式、监听应用生命周期
├─pages.json         配置页面路由、导航栏、tabBar等页面类信息
├─manifest.json      配置appid、应用名称、logo、版本等打包信息
└─uni.scss           uni-app内置的常用样式变量
```

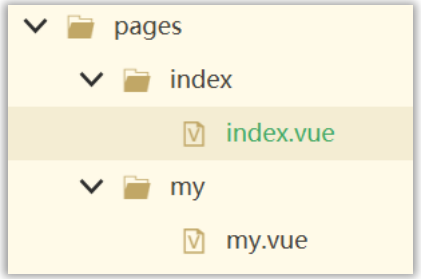


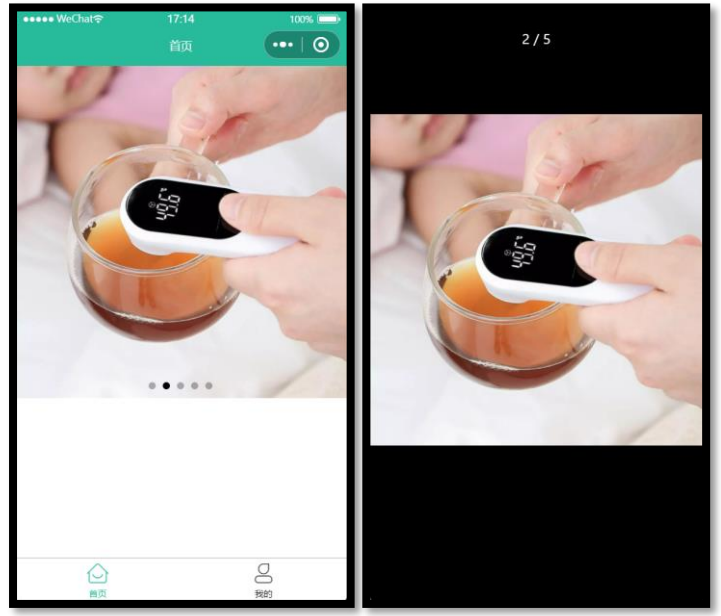


04 uni-app和原生小程序开发区别

uni-app 和 原生小程序开发区别

每个页面是一个 .vue 文件，数据绑定及事件处理同 Vue.js 规范：





1. 属性绑定　src="{{ url }}" 升级成 :src="url"
2. 事件绑定　bindtap="eventName" 升级成 @tap="eventName"，支持（）传参
3. 支持 Vue 常用指令　v-for、 v-if、v-show、v-model 等

温馨提示：调用接口能力，**建议**前缀 wx 替换为 uni，养成好习惯，这样支持多端开发。

命令行创建 uni-app 项目

命令行创建 uni-app 项目

uni-app 支持两种方式创建项目：

- 通过HBuilderX 创建
- 通过命令行创建（不必依赖 HBuilderX）

命令行创建 uni-app 项目：

vue3+ts版：npx degit dcloudio/uni-preset-vue#vite-ts 项目名称

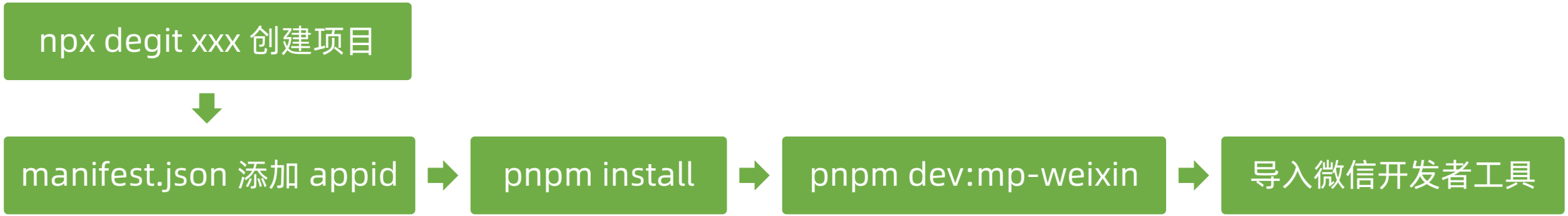
官网链接：https://uniapp.dcloud.net.cn/quickstart-cli.html#创建uni-app

命令行创建 uni-app 项目

uni-app 支持两种方式创建项目：

- 通过HBuilderX 创建
- 通过命令行创建（不必依赖 HBuilderX）

编译和运行 uni-app 项目：

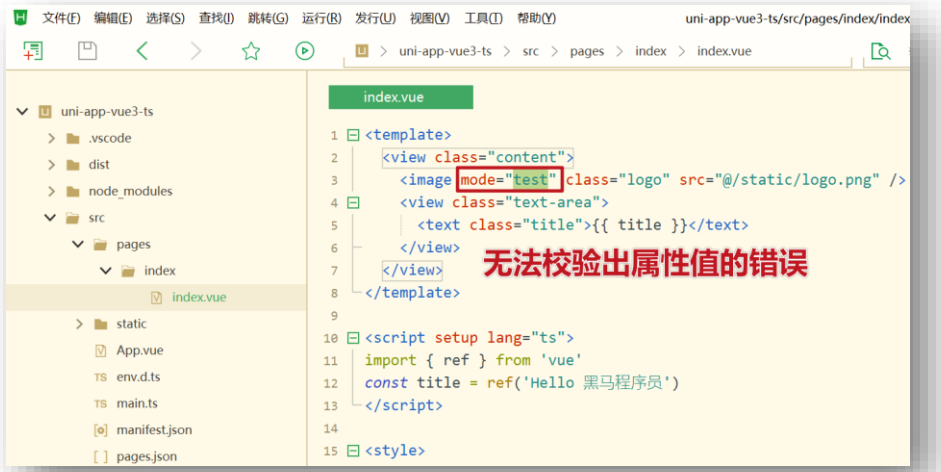


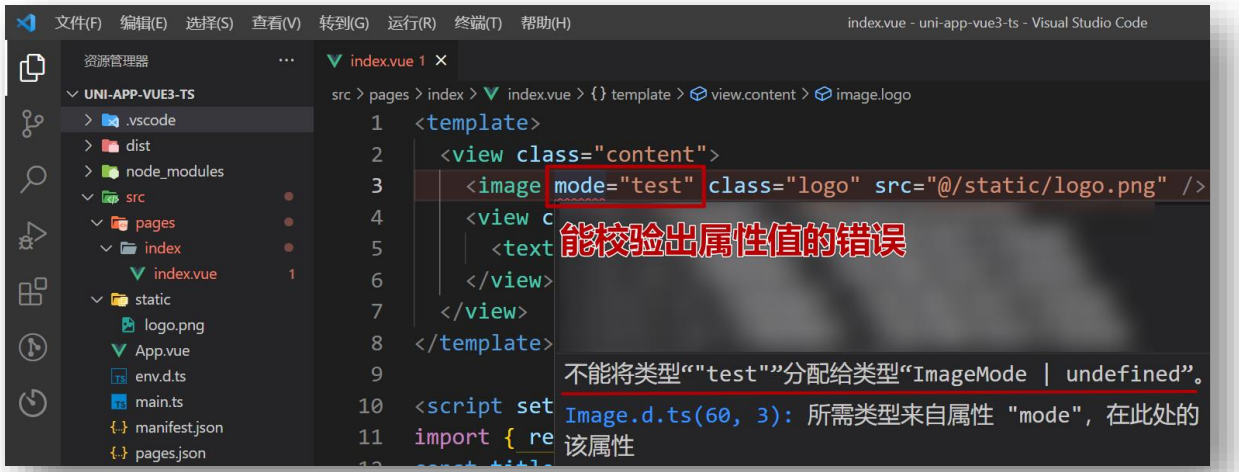
06 用 VS Code 开发 uni-app 项目

用 VS Code 开发 uni-app 项目

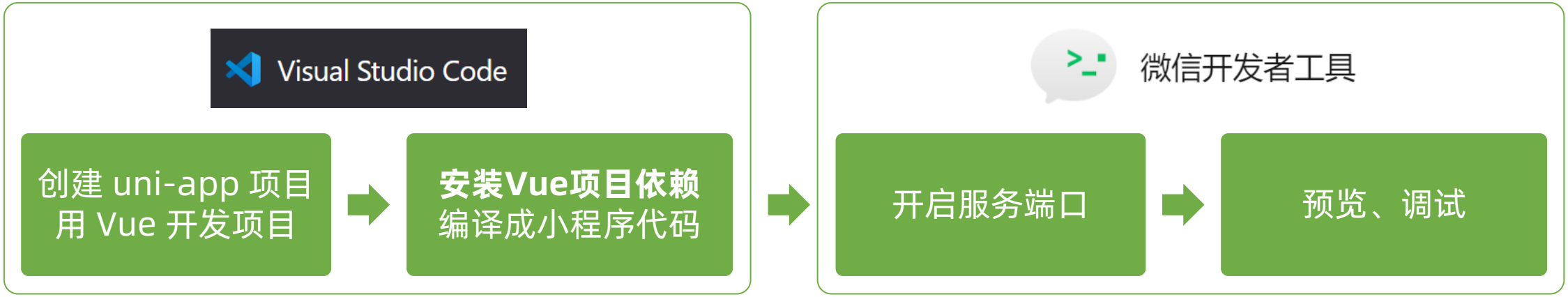
为什么选择 VS Code ?

- HbuilderX 对 TS 类型支持暂不完善
- VS Code 对 TS 类型支持友好，熟悉的编辑器





用 VS Code 开发 uni-app 项目



用 VS Code 开发 uni-app 项目

安装 uni-app 插件

- 快速创建页面
- uni-app 代码提示
- 鼠标悬停查文档

ts 类型校验

- 安装类型声明文件
- 配置 tsconfig.json

```
pnpm i -D @types/wechat-miniprogram @uni-helper/uni-app-types
```

```
{
  "compilerOptions": {
    "types": [
      "@dcloudio/types",
+       "@types/wechat-miniprogram",
+       "@uni-helper/uni-app-types"
    ]
  },
+  "vueCompilerOptions": {
+    "nativeTags": ["block", "component", "template", "slot"]
+  }
}
```

json 注释问题

07 拉取小兔鲜儿项目模板代码

拉取小兔鲜儿项目模板代码

git clone http://git.itcast.cn/heimaqianduan/erabbit-uni-app-vue3-ts.git heima-shop

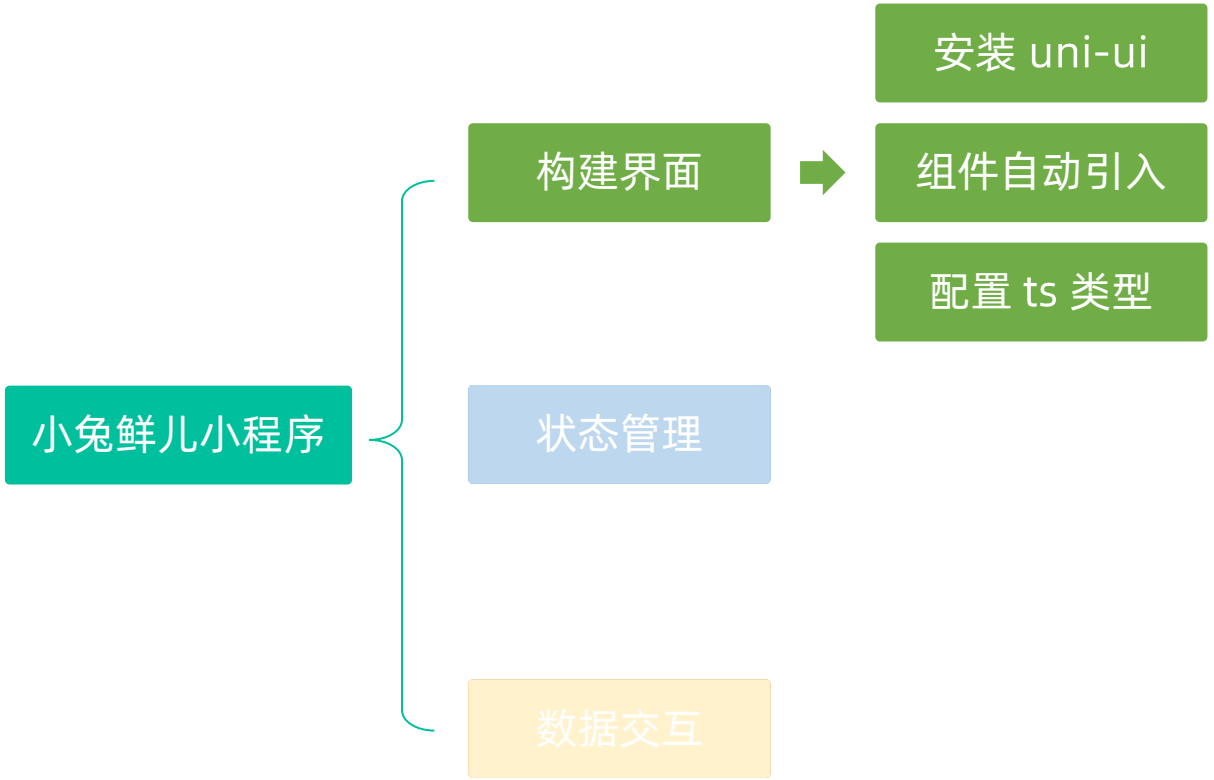
注意事项

- 在 manifest.json 中添加微信小程序的 appid

pnpm install ➡ pnpm dev:mp-weixin ➡ 导入微信开发者工具

基础架构 – 引入 uni-ui 组件库

基础架构 - 使用 uni-ui 组件库




```
pnpm i @dcloudio/uni-ui
```

```
// pages.json
{
  "easycom": {
    "autoscan": true,
    "custom": {
      // uni-ui 规则如下配置
      "^uni-(.*)": "@dcloudio/uni-ui/lib/uni-$1/uni-$1.vue"
    }
  },
  "pages": [
    // ...
  ]
}
```

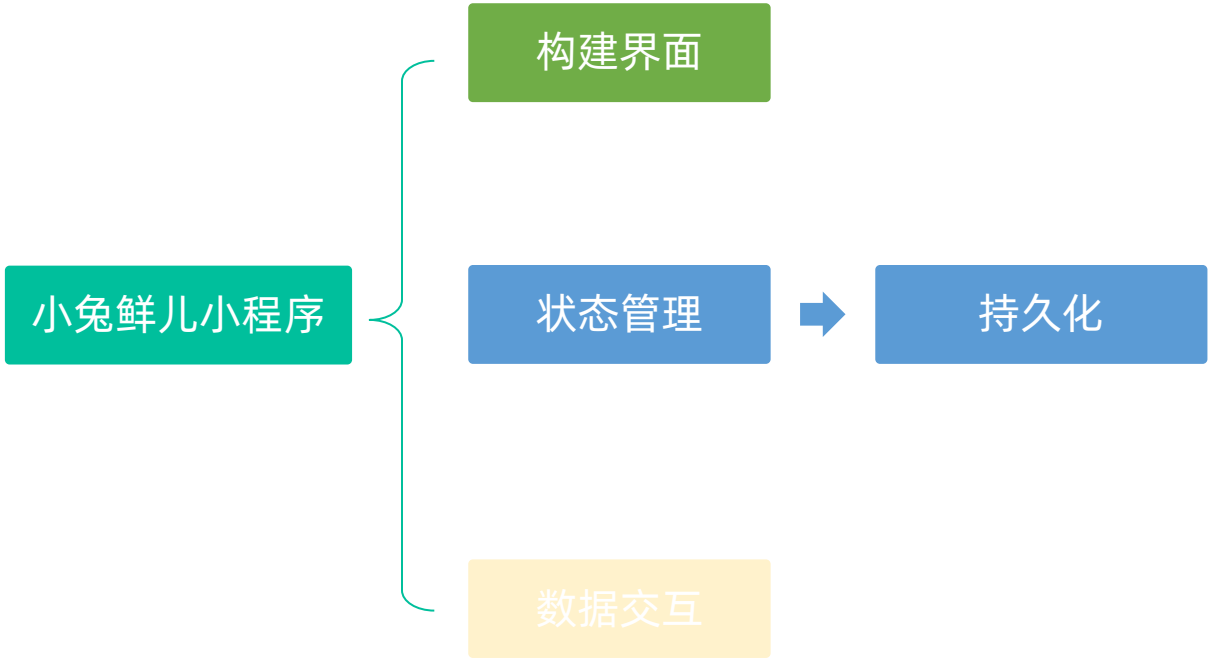
```
pnpm i -D @uni-helper/uni-ui-types
```

```
// tsconfig.json
{
  "compilerOptions": {
    "types": [
      "@dcloudio/types",
      "@types/wechat-miniprogram",
      "@uni-helper/uni-app-types",
+     "@uni-helper/uni-ui-types"
    ]
  },
}
```

官方文档

基础架构 – 小程序端 Pinia 持久化

基础架构 - 小程序端 Pinia 持久化




```
// 网页端API
localStorage.setItem()
localStorage.getItem()
```

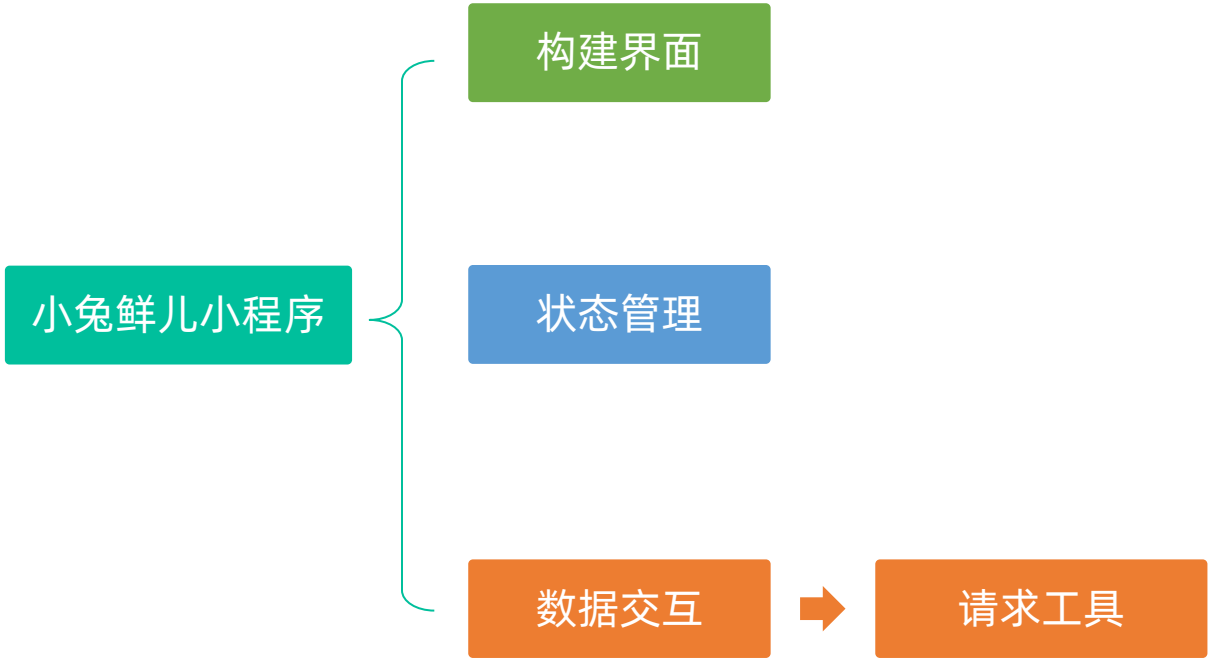
```
// 兼容多端API
uni.setStorageSync()
uni.getStorageSync()
```

```
// stores/modules/member.ts
export const useMemberStore = defineStore(
  'member',
  () => {
    //…省略
  },
  {
    // 配置持久化
    persist: {
      // 调整为兼容多端的API
      storage: {
        setItem(key, value) {
          uni.setStorageSync(key, value)
        },
        getItem(key) {
          return uni.getStorageSync(key)
        },
      },
    },
  }
)
```

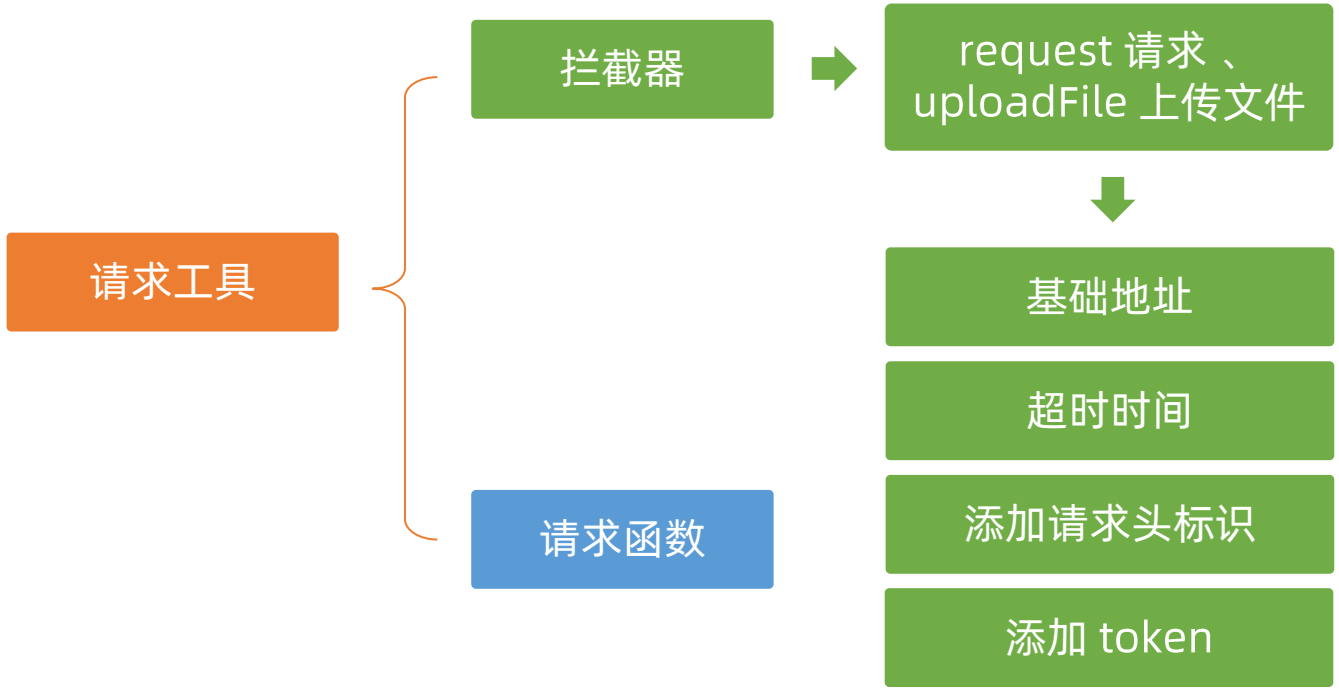
官方文档

基础架构 – uni.request 请求封装

基础架构 - 请求和上传文件拦截器



基础架构 - 请求和上传文件拦截器



拦截器文档

uni.addInterceptor(STRING, OBJECT)

添加拦截器

STRING 参数说明

需要拦截的 `api` 名称，如：`uni.addInterceptor('request', OBJECT)` ，将拦截 `uni.request()`

OBJECT 参数说明

参数名	类型	必填	默认值	说明	平台差异说明
invoke	Function	否		拦截前触发	

接口文档

说明

服务器

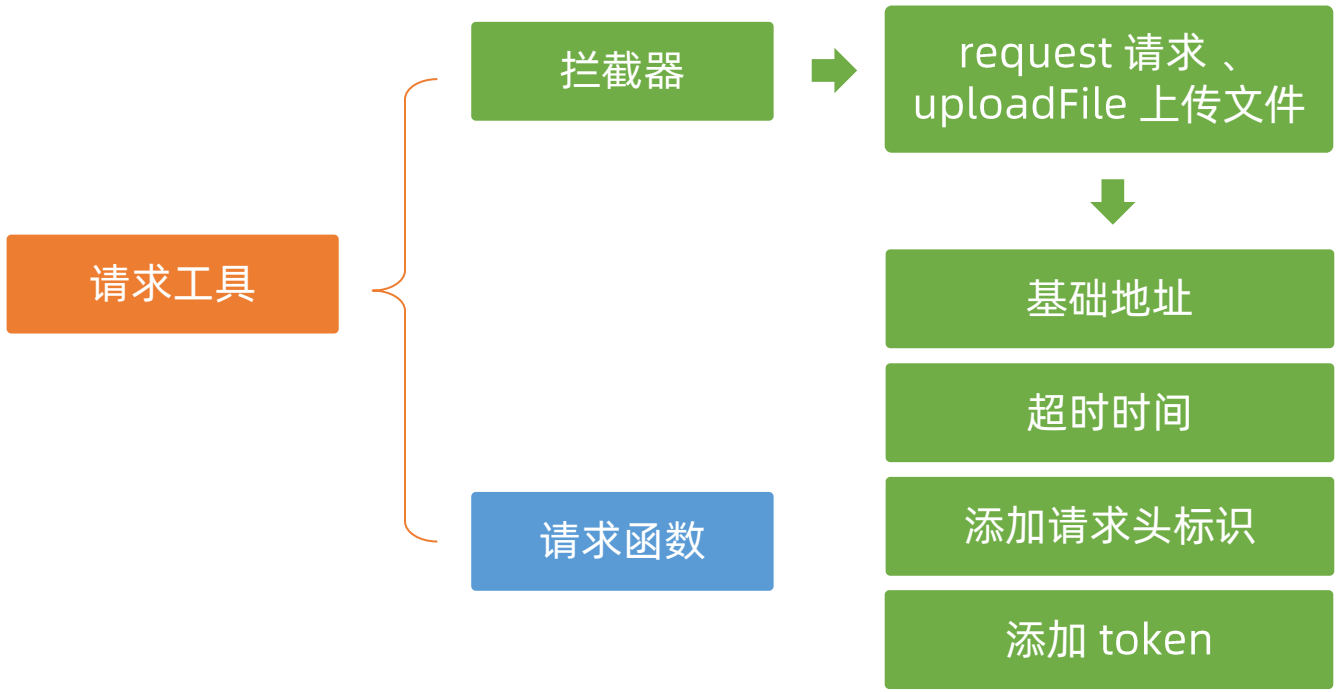
- 首选服务器：https://pcapi-xiaotuxian-front-devtest.itheima.net
- 备用服务器：https://pcapi-xiaotuxian-front.itheima.net

请求参数

- 小程序端调用，请求头中 header 中添加：`'source-client': 'miniapp'`
- 用户登录成功后，调用需要 token 的接口，无 token 或者 token 错误，响应状态码是 401

参数名	位置	类型	必填	说明
source-client	header	string	是	示例值：miniapp 代表小程序端，app 代表 App 端
Authorization	header	string	否	示例值：`Authorization: token`

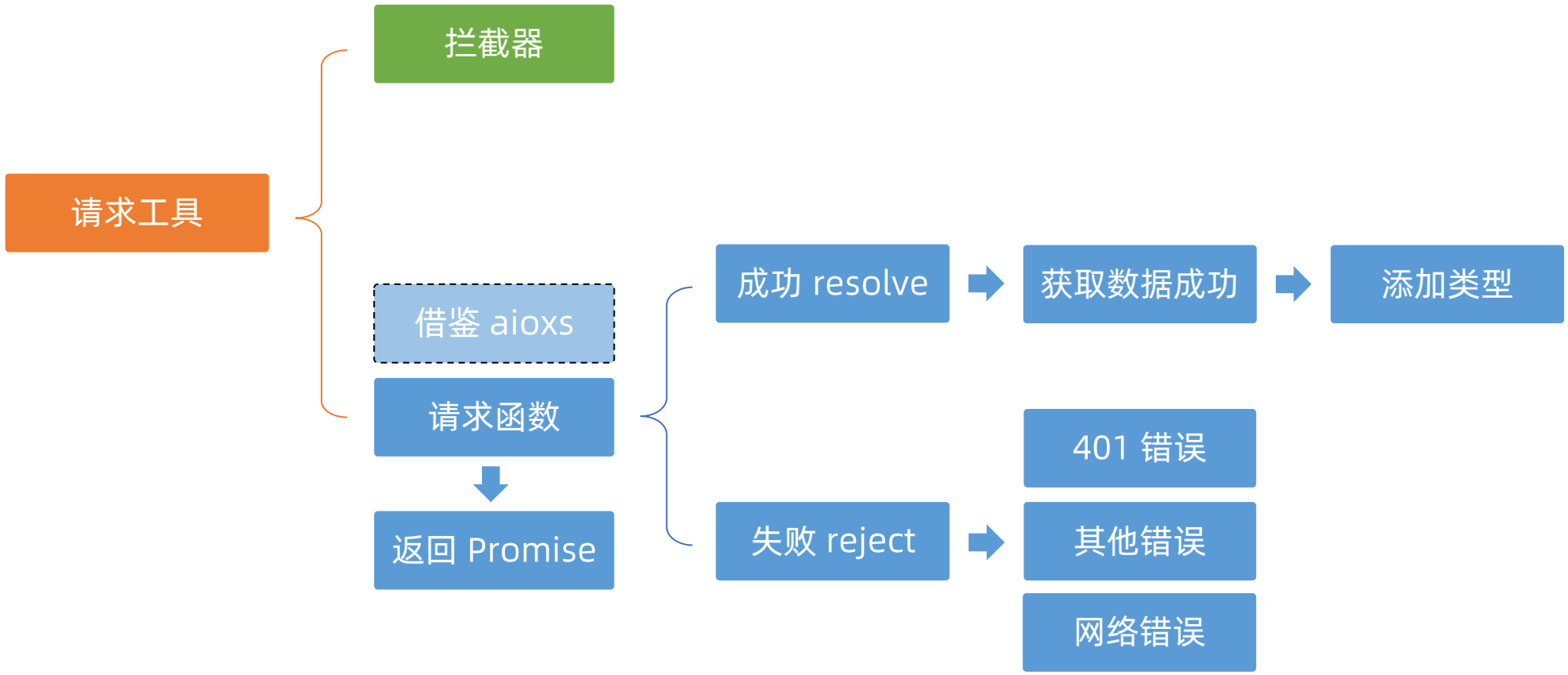
基础架构 - 请求和上传文件拦截器




```
// 拦截 request 请求
uni.addInterceptor('request', httpInterceptor)
// 拦截 uploadFile 文件上传
uni.addInterceptor('uploadFile', httpInterceptor)
```

```
const httpInterceptor = {
  // 拦截前触发
  invoke(options: UniApp.RequestOptions) {
    // 1. 非 http 开头需拼接地址
    if (!options.url.startsWith('http')) {
      options.url = baseURL + options.url
    }
    // 2. 请求超时
    options.timeout = 10000
    // 3. 添加小程序端请求头标识
    options.header = {
      ...options.header,
      'source-client': 'miniapp',
    }
    // 4. 添加 token 请求头标识
    const memberStore = useMemberStore()
    const token = memberStore.profile?.token
    if (token) {
      options.header.Authorization = token
    }
  }
}
```

基础架构 - 封装 Promise 请求函数



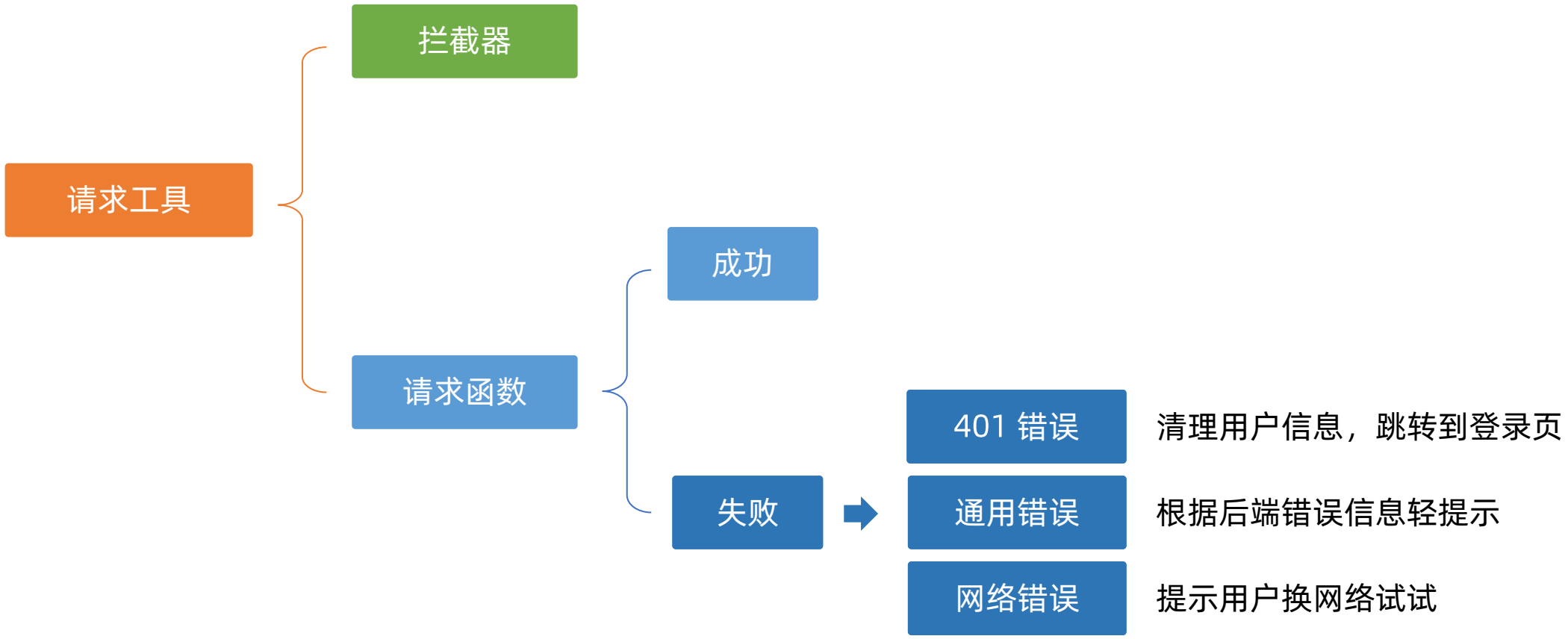
请求函数 - 请求成功提取数据和设置类型

```ts
const res = await http<BannerItem[]>({
  method: 'GET',
  url: '/home/banner',
})
```

```ts
export const http = <T>(options: UniApp.RequestOptions) => {
  return new Promise<Data<T>>((resolve, reject) => {
    uni.request({
      ...options,
      success(res) {
        resolve(res.data as Data<T>)
      },
    })
  })
}
```

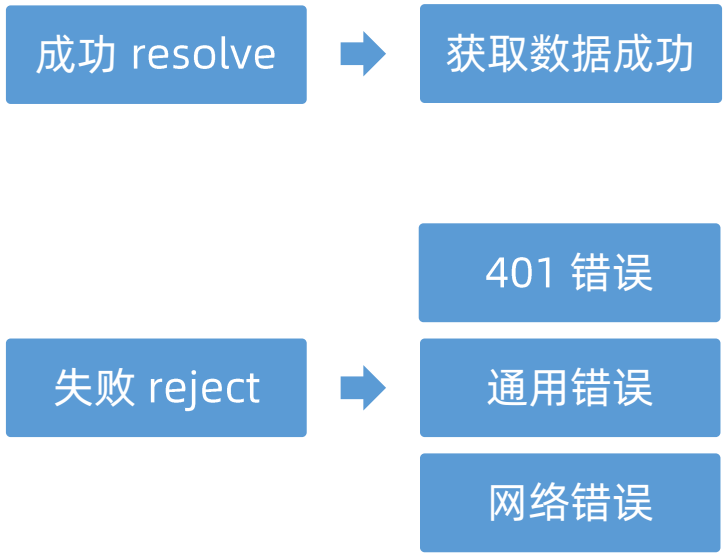
```ts
interface Data<T> {
  code: string
  msg: string
  result: T
}
```

基础架构 - 封装 Promise 请求函数



请求函数 - 获取数据失败

- uni.request 的 success 回调函数只是表示服务器响应成功，没处理响应状态码，业务中使用不方便
- axios 函数是只有响应状态码是 2xx 才调用 resolve 函数，表示获取数据成功，业务中使用更准确

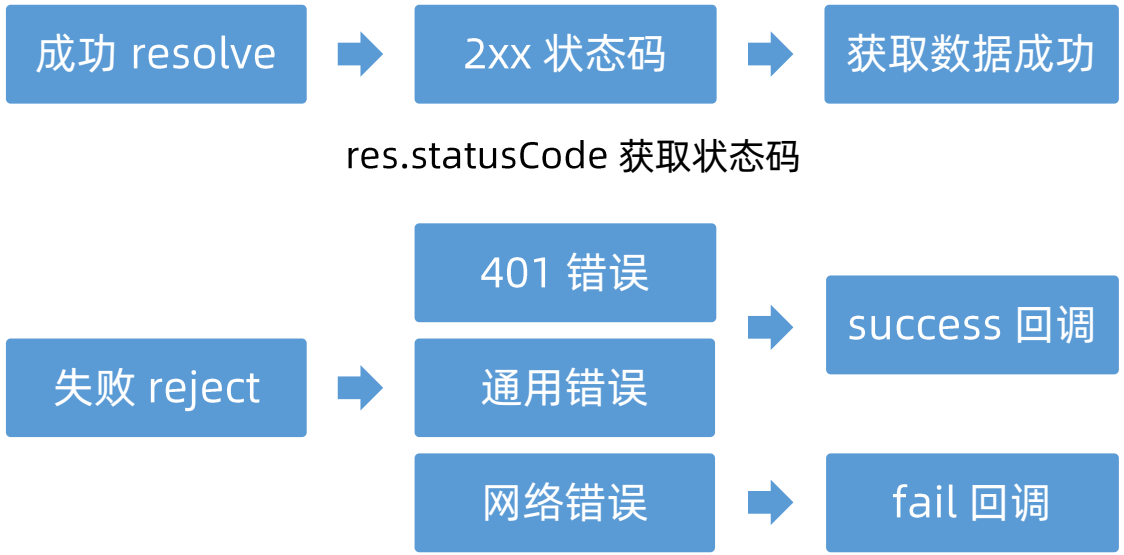



```
uni.request({
  ...options,
  // 响应成功
  success(res) {

  },
  // 响应失败
  fail(err) {

  },
})
```

```
const res = await http<BannerItem[]>({
  method: 'GET',
  url: '/home/banner',
})
console.log('获取数据成功', res.result)
```

请求函数 - 获取数据失败

- uni.request 的 success 回调函数只是表示服务器响应成功，没处理响应状态码，业务中使用不方便
- axios 函数是只有响应状态码是 2xx 才调用 resolve 函数，表示获取数据成功，业务中使用更准确




```
uni.request({
  ...options,
  // 响应成功
  success(res) {
    if (res.statusCode >= 200 && res.statusCode < 300) {
      // 获取数据成功，调用resolve
    } else if (res.statusCode === 401) {
      // 401错误，调用reject
    } else {
      // 通用错误，调用reject
    }
  },
  // 响应失败
  fail(err) {
    // 网络错误，调用reject
  },
})
```

请求函数 - 获取数据失败

成功 resolve ➡ 2xx 状态码

```
// 响应成功
success(res) {
  // 2xx 状态码 -> 获取数据成功
  if (res.statusCode >= 200 && res.statusCode < 300) {
    resolve(res.data as Data<T>)
  }
  // 其他状态码 -> 获取数据失败
```

失败 reject ➡ 401 错误 / 通用错误 / 网络错误

```
  else if (res.statusCode === 401) {
    const memberStore = useMemberStore()
    memberStore.clearProfile()
    uni.navigateTo({ url: '/pages/login/login' })
    reject(res)
  }
```

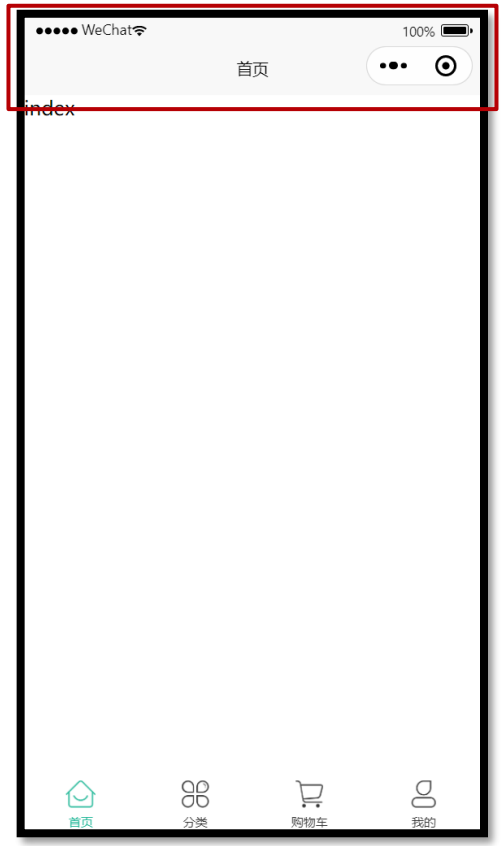
```
  else {
    uni.showToast({
      icon: 'none',
      title: (res.data as Data<T>).msg || '数据获取失败',
    })
    reject(res)
  }
```


```
// 响应失败
fail(err) {
  uni.showToast({ icon: 'none', title: '网络错误，换个网络试试' })
  reject(err)
},
```

首页 – 自定义导航栏

首页 - 自定义导航栏

默认导航栏



自定义导航栏

样式适配

首页 - 自定义导航栏

1. 准备组件
2. 隐藏默认导航栏，修改文字颜色
3. 样式适配 -> 安全区域

iPhone 6/7/8　　iPhone X/11/12/13　　iPhone 14 Pro Max

```
// pages.json
{
  "path": "pages/index/index",
  "style": {
    "navigationStyle": "custom",
    "navigationBarTextStyle": "white",
    "navigationBarTitleText": "首页"
  }
},
```

```
// 获取屏幕边界到安全区域距离
const { safeAreaInsets } = uni.getSystemInfoSync()
```

```
<template>
  <view
    class="navbar"
    :style="{ paddingTop: safeAreaInsets?.top + 'px' }"
  >
    ...省略
  </view>
</template>
```

传智教育旗下高端IT教育品牌