

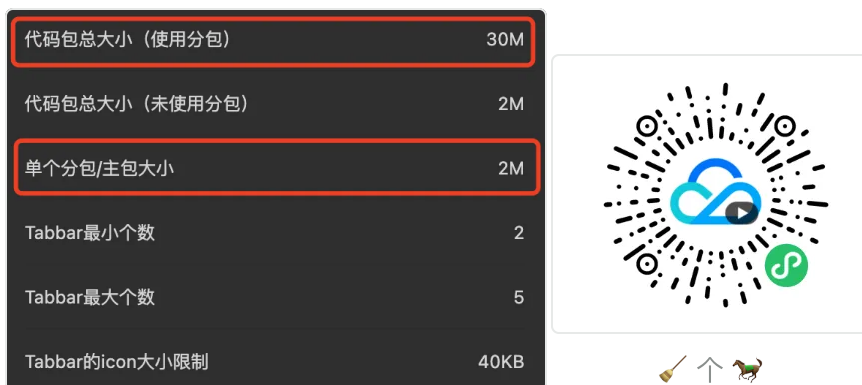
# 包体积&分包

---

- 1. 为什么要关注包体积
- 2. 主包
- 3. 分包
  - 3.1. 分包扩展功能
  - 3.2. 打包结果
  - 3.3. 使用
- 4. 微信小程序如何配置分包
  - 4.1. 注册分包
  - 4.2. 构建 npm 配置
  - 4.3. 分包产物
  - 4.4. 如何跳转到分包
- 5. uni-app 如何配置分包
  - 5.1. 注册分包
  - 5.2. 处理分包的依赖
  - 5.3. 构建 npm
  - 5.4. 编译运行到微信小程序
- 6. 参考资料

背景：最近需要重构腾讯云音视频小程序，第一担心事情包体积会不会超，会不会影响发布。

## 1. 为什么要关注包体积

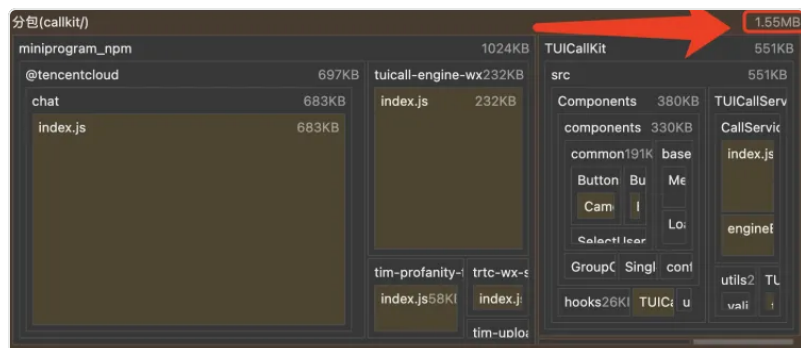


- 如上图所示，微信小程序的总包体积不能超过 30M, **单个包最多 2M, 超过了小程序就没办法发布。**
- 体积概念：

类别	大小
一个模糊的 png 图片	 <p>avatar.png PNG 图像 - 6 KB</p>
一个超级简单的个人中心页面。	<div>  <div>  <p>4个项目 4份文稿 - 14 KB</p> </div> </div>
一个被压缩后的 2s 音频文件	 <p>9KB</p>

callkit 体积

(这里是主包进行了部分分担后的体积)



1.5 M

## 2. 主包

- 主包：放置默认启动页面&TabBar 页面，以及一些所有分包都需用到公共资源/JS 脚本（💡：除 subPackages 配置之外的所有文件）。
- 分包：subPackages 里边的文件。
- 主包体积过大，会影响小程序的首次启动体验（代码下载、代码注入、渲染等）。
- 主包体积限制为 2M，当代码体积超过 2M 时，需要进行分包处理，否则无法发布。
- 备注：
  - tabbar 页面必须放在主包。
  - 即使是分包下的 assets 资源，也会被打入主包。

## 3. 分包

- 分包：放在 subPackages 里边的文件。
- 分包的好处：
  - 解决主包体积过大时不能发布小程序的问题。
  - 解决主包过大，首次启动小程序体验卡断等问题。
- 分包体积限制：
  - 整个小程序所有分包大小不超过 30M（服务商代开发的小程序不超过 20M，箭指 uni-app）
  - 单个分包/主包大小不能超过 2M。
- 分包的坏处：
  - 无法进行全局监听。

### 3.1. 分包扩展功能

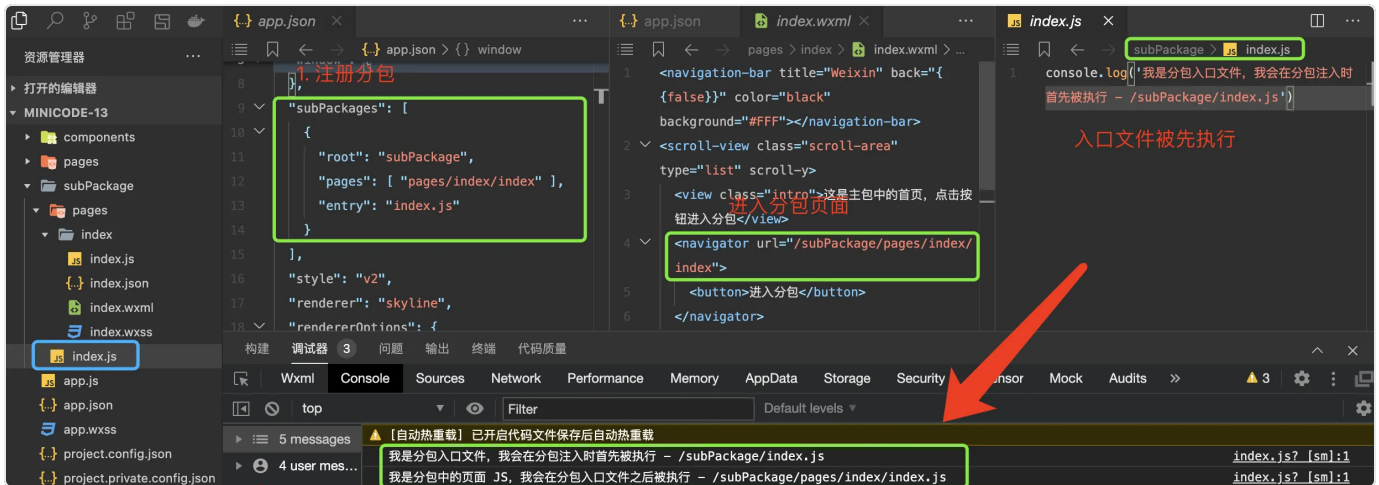
- 独立分包：
  - 从独立分包进入小程序，不需要下载主包。
  - 独立于主包和其它分包运行，因此不能依赖主包和其它分包内容，getApp() 也不一定获得。
  - 一个小程序可以有多个独立分包。
  - 通过 `independent` 属性配置。
- 分包预下载：解决首次进入分包页面加载缓慢等问题。通过在 `app.json` 增加 `preloadRule` 配置来控制。
- 分包异步化：分包本来只能将页面进行处理，通过分包异步化，可以将主包的插件、代码逻辑、组件等剥离到分包；更有效降低分包体积。

### 3.2. 打包结果

打包原则： `subPackages` 配置路径外的目录将被打包到主包中。

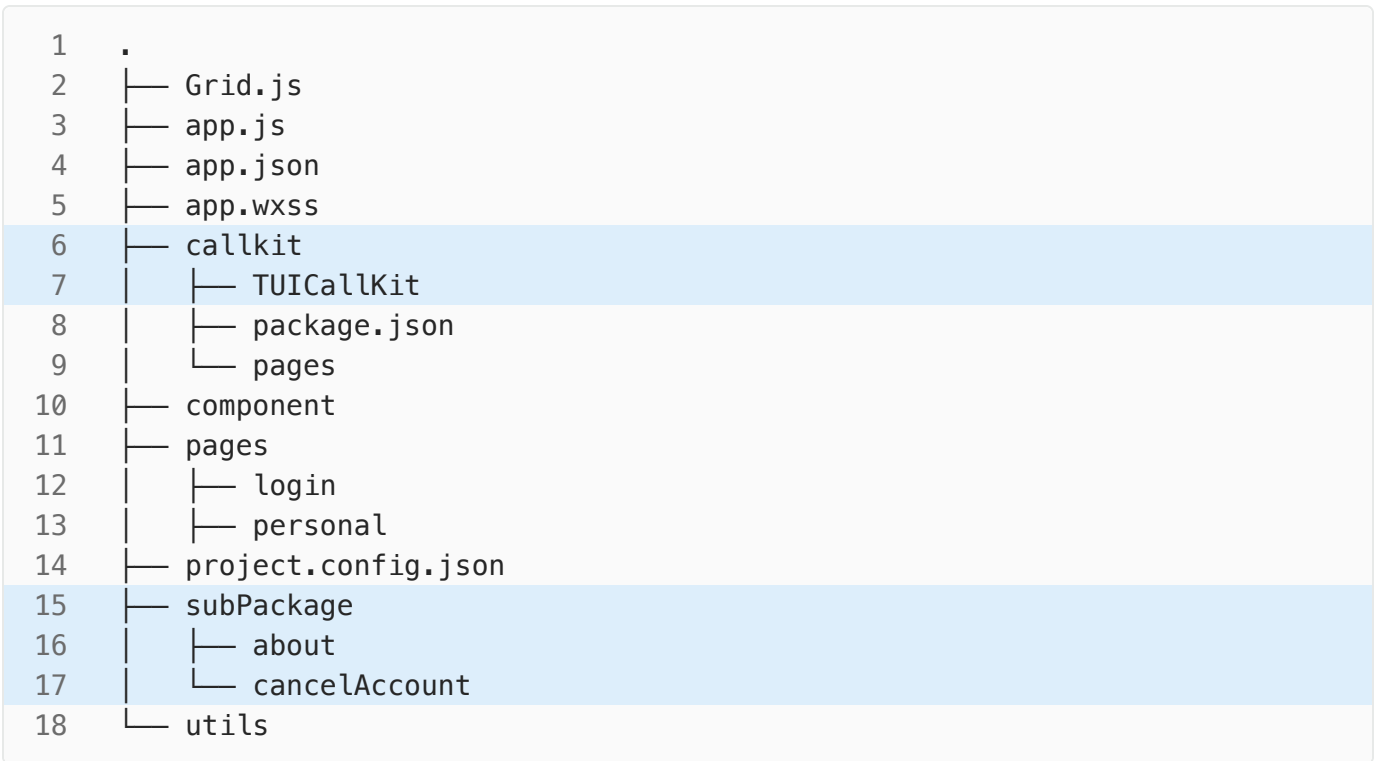


### 3.3. 使用



## 4. 微信小程序如何配置分包

假如我需要把 callkit roomkit subPackage 三个目录放入分包（分包的目录名可以随便定义）。



### 4.1. 注册分包

- app.json 中配置要放入分包的文件路径。

```

1  // app.json
2  {
3    "pages": [
4      "pages/login/login",
5      "pages/personal/personal"
6    ],
7    "subPackages": [
8      {
9        "root": "subPackage",
10       "pages": [
11         "cancelAccount/cancelAccount",
12         "about/about"
13       ]
14     },
15     {
16       "root": "callkit",
17       "pages": [
18         "pages/call"
19       ]
20     },
21   ],
22 }

```

- subPackage 目录的分包就配置好了。

## 4.2. 构建 npm 配置

- 如果你的分包中的页面，依赖了一些第三方包，那么需要在分包中下载，构建 npm；
- 如果在主包下载、构建，那么这部分第三方包的体积会被打入主包。
- eg. callkit 这个分包依赖了一些第三方包，@tencentcloud/chat@latest ...

```

1  |— callkit
2  |   |— TUICallKit
3  |   |— pages

```

- 这里的 callkit 依赖了一些第三方包，那么需要 执行如下命令。执行 后的产物：  
callkit/node\_modules

```

1  cd callkit
2  npm init -y
3  # 下载依赖的包, 这里以 callkit 依赖的为例
4  npm i @tencentcloud/chat@latest
5  npm i @tencentcloud/tui-core@latest
6  npm i tuicall-engine-wx@2.x

```






- 配置构建 npm 的目录, 如果不配置, 这里下载的内容, 会被无法构建; 或者构建进去了主包。(配置的好处, 不用分别进入分包取执行构建 npm)。

```

1  "setting": {
2    "urlCheck": false,
3    "es6": true,
4    "postcss": false,
5    "minified": false,
6    "newFeature": true,
7    "bigPackageSizeSupport": true,
8    "packNpmManually": true,
9    "packNpmRelationList": [
10   {
11     "packageJsonPath": "./package.json",
12     "miniprogramNpmDistDir": "./"
13   },
14   {
15     "packageJsonPath": "./callkit/package.json",
16     "miniprogramNpmDistDir": "./callkit"
17   }
18 ],
19 }

```

- 构建后的产物。callkit/miniprogram\_npm

< 代码包 / 分包(callkit/)		大小
 miniprogram_npm		1024KB
 TUICallKit		551KB
 pages		9KB
 package-lock.json		6KB
 package.json		356B

### 4.3. 分包产物

主包	1.18MB
分包(roomkit/)	1.87MB
分包(callkit/)	1.55MB
分包(subPackage/)	7KB

## 4.4. 如何跳转到分包

```
1 uni.navigateTo({ url: '/callkit/pages/call' });
```

# 5. uni-app 如何配置分包

## 5.1. 注册分包

```
1 // pages.json
2 {
3   "pages": [
4     "pages/login/login",
5     "pages/personal/personal"
6   ],
7   "subPackages": [
8     {
9       "root": "subPackage",
10      "pages": [
11        "cancelAccount/cancelAccount",
12        "about/about"
13      ]
14    },
15    {
16      "root": "callkit",
17      "pages": [
18        "pages/call"
19      ]
20    },
21  ],
22 }
```

## 5.2. 处理分包的依赖



- 将分包中的依赖在 vite.config.ts 配置到一下，防止被打入主包。

```
1 optimizeDeps: {
2   include: [
3     'tuicall-engine-wx',
4     '@tencentcloud/chat',
5     '@tencentcloud/tui-core',
6   ],
7 }
```

```
1 build: {
2   rollupOptions: {
3     external: [
4       'tuicall-engine-wx',
5       '@tencentcloud/chat',
6       '@tencentcloud/tui-core',
7     ],
8   },
9 }
```

### 5.3. 构建 npm

- 在 manifest.json 文件开启分包，如果需要构建 npm，同时进行配置。

```
1 "mp-weixin" : {
2   "appid": "",
3   "setting": {
4     "urlCheck": false,
5     "packNpmManually": true,
6     "packNpmRelationList": [
7       {
8         "packageJsonPath": "./package.json",
9         "miniProgramNpmDistDir": "./"
10      },
11      {
12        "packageJsonPath": "./callkit/package.json",
13        "miniProgramNpmDistDir": "./callkit"
14      }
15    ]
16   "usingComponents": true,
17   "optimization": {
18     "subPackages": true
19   }
20 }
```

## 5.4. 编译运行到微信小程序

```
1 # 编译运行到小程序
2 npm run dev:mp-weixin
```

```
1 # 启动一个新的命令行，下载分包依赖
2 cd callkit
3 npm init -y
4 npm i @tencentcloud/chat@latest
5 npm i @tencentcloud/tui-core@latest
6 npm i tuicall-engine-wx@2.x
```

- 构建 npm。

## 6. 参考资料

该资料参考日期：2024.11.05

- 微信小程序：
  - npm 支持：<https://developers.weixin.qq.com/miniprogram/dev/devtools/npm.html>
  - 包体积优化：  
[https://developers.weixin.qq.com/miniprogram/dev/framework/performance/tips/start\\_optimizeA.html](https://developers.weixin.qq.com/miniprogram/dev/framework/performance/tips/start_optimizeA.html)
  - 使用分包：  
<https://developers.weixin.qq.com/miniprogram/dev/framework/subpackages/basic.html>
- uni-app 小程序：
  - 分包配置：<https://uniapp.dcloud.net.cn/collocation/pages.html#subpackages>
  - mp-weixin 配置：<https://uniapp.dcloud.net.cn/collocation/manifest.html#mp-weixin>