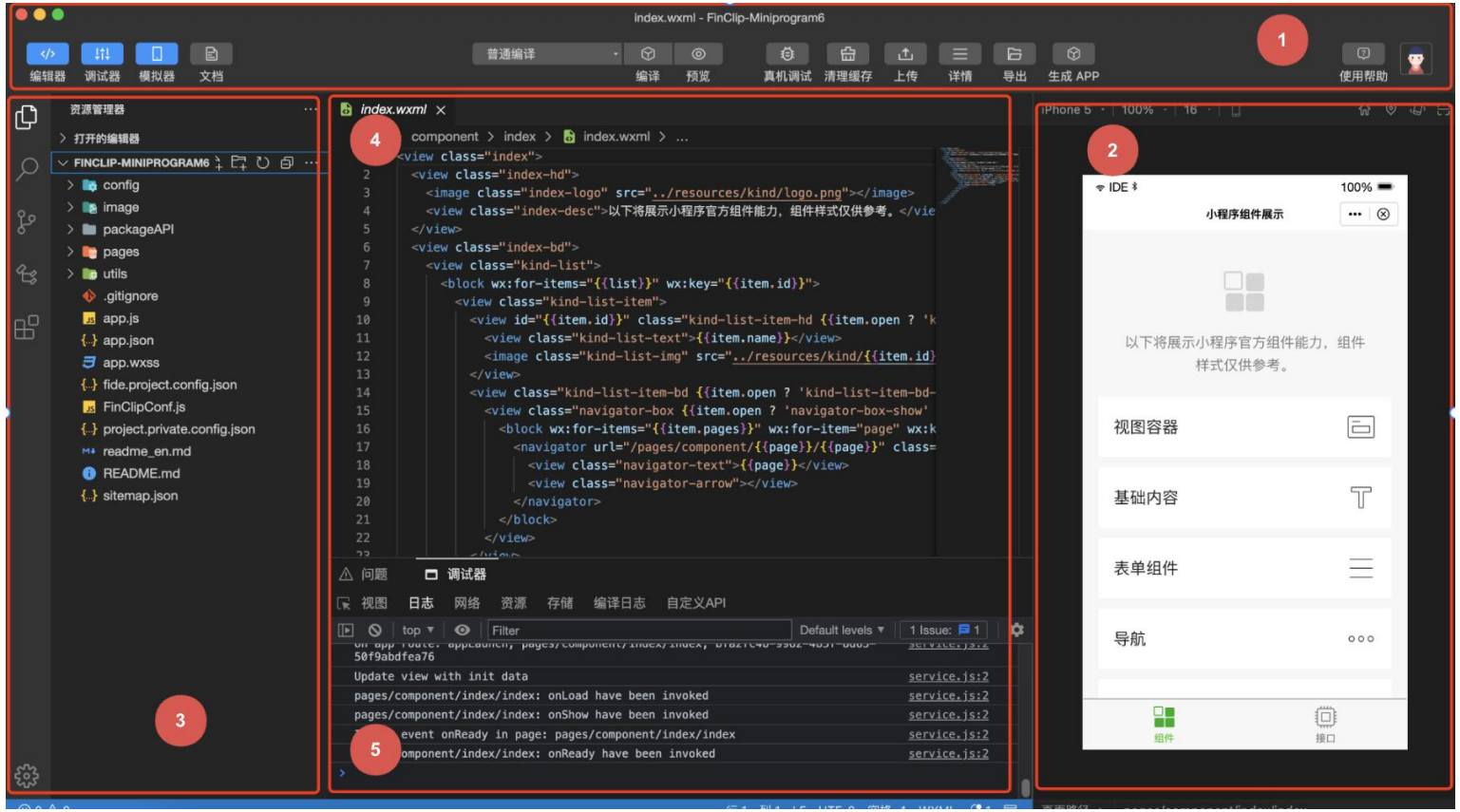
睿智开元

开元小程序开放平台
小程序开发培训

上海锐图智能科技有限公司

开发工具：睿制开元

- 工具栏：可以展示关闭不同面板，打开文档中心，进行编译或其他操作
- 模拟器：可以模拟预览小程序的使用效果，也可以设置不同的机型和显示比例查看调整预览效果
- 文件管理器：可以查看目录数结构与内容，新增文件或文件夹。点击任意文件都可以在右侧看到文档中的具体内容，与内容编辑器被统称为「编辑器」
- 内容编辑器：可以修改代码内容，与文件管理器被统称为「编辑器」
- 调试器：可以调试小程序代码，查看报错或其他提示



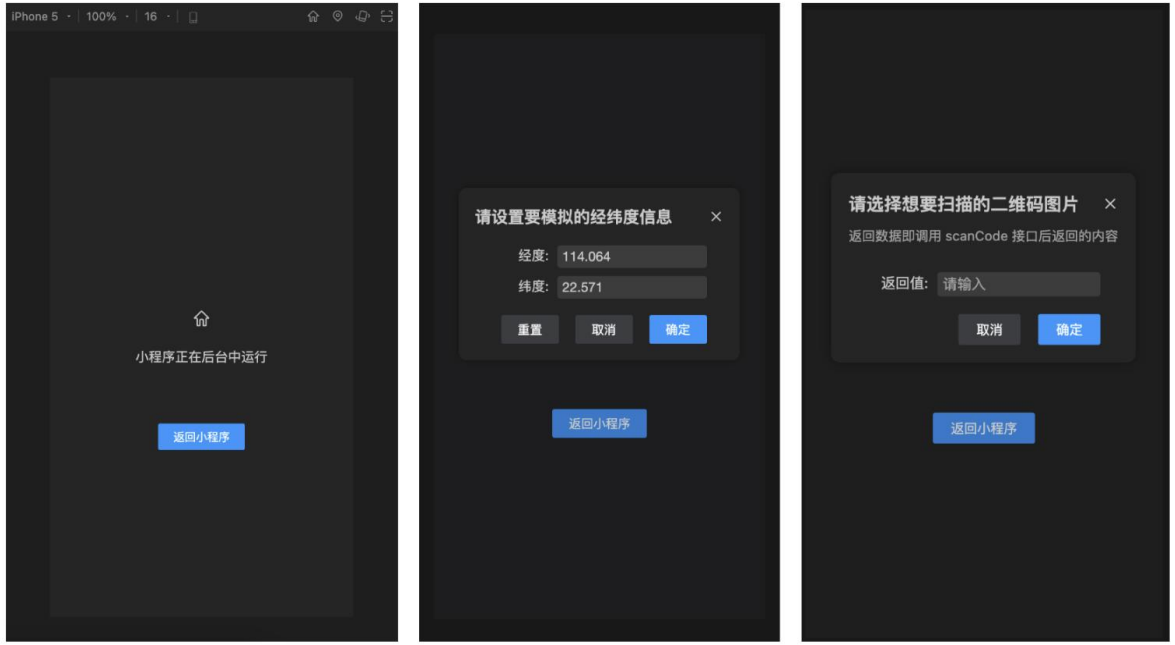
工具栏

- 模拟器、编辑器、调试器：切换下方对应面板的开关
- 文档：会跳转并打开「Finclip Studio」操作指引文档
- 切换编译模式：配置对应的编译模式（如配置启动页面，参数），以及进行编译与预览操作
- 真机调试：让开发者在真机中调试小程序，所有的操作信息都会被传回 Finclip Studio中
- 清缓存：模拟清除缓存后小程序调试（支持清除文件缓存/清除编译缓存/全部清除3 种模式）
- 上传：则是将当前的小程序代码包，上传到所属小程序的版本目录中
- 详情：查看小程序基础库版本，小程序 appid 、备案的开发域名信息等
- 导出：则可以将当前的小程序导出成 .zip 格式的代码包
- 生成 App：将小程序转换生成对应的 App 工程或安装文件

模拟器

模拟器可以反应 FinClip 小程序在客户端的真实逻辑表现，绝大多数的 Api 都可以在模拟器中展示正确的状态。您可以通过左上角的按钮切换机型与分辨率，调试在不同尺寸机型中小程序的适配问题。

此外还可以通过右上角的操作按钮分别对小程序进行「切后台/模拟定位/摇一摇/扫码」的模拟操作，从而对小程序中的相关业务逻辑进行模拟测试。



文件管理器

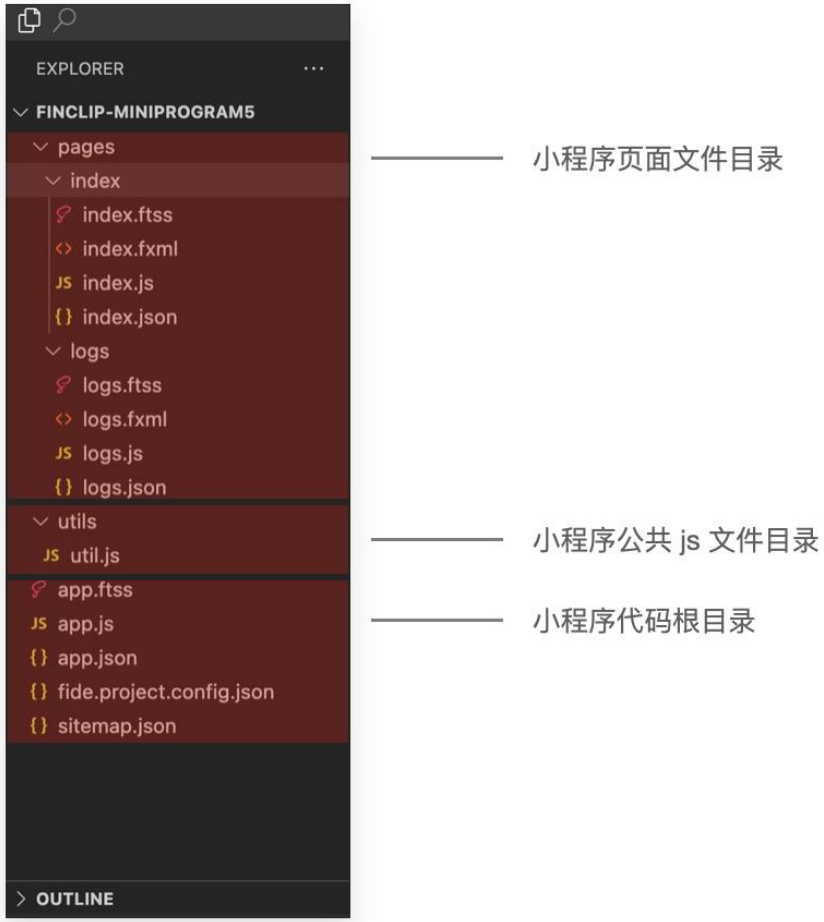
文件管理器中可以帮助您查看小程序相关的文件，我们可以在「小程序代码根目录」下看到一共 5 个小程序应用级别的文件，分别是：

- app.ftss 小程序全局公共样式文件
- app.js 小程序逻辑文件
- app.json 小程序配置文件
- fide.project.config.json 项目配置文件
- sitemap.json 小程序索引 sitemap

「pages」 文件夹是用来存储小程序中每个页面的具体目录。

index.ftss 页面样式文件

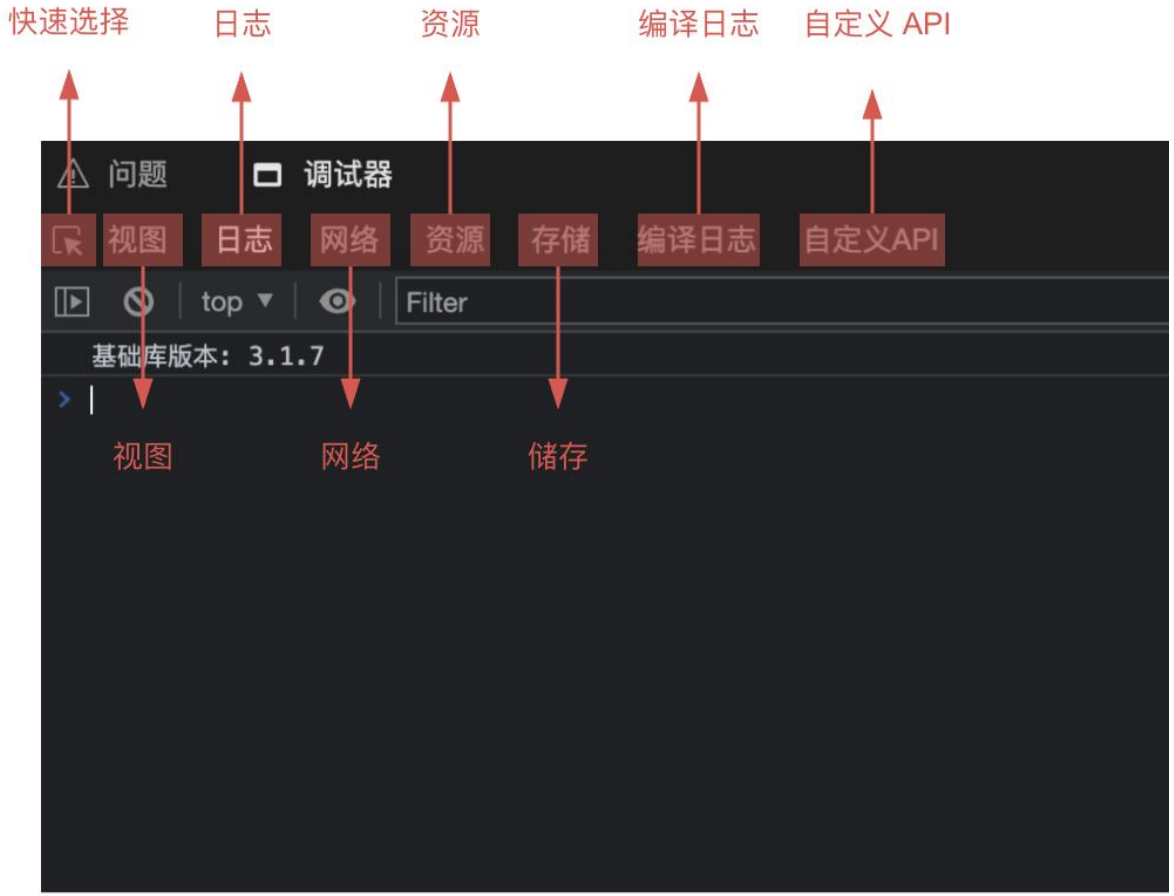
- index.fxml 页面结构文件
- index.js 页面逻辑文件
- index.json 页面配置文件



调试器

调试器包含如下功能模块：

- 快速选择：可以快速定位页面中组件对应的 ftml 代码
- 视图：查看真实的页面结构与结构对应的 ftss 属性，同时可以通过修改对应 ftss 属性，在模拟器中实时看到修改的情况
- 日志：开发者可以在此调试代码
- 网络：用于观察和显示 request 和 socket 的请求情况
- 存储：用于展示以及修改当前 page 的 data 属性
- 编译日志：用于观察和显示 FinClip 小程序的编译过程与报错信息
- 自定义 API ：用于修改测试 Api 的回调内容



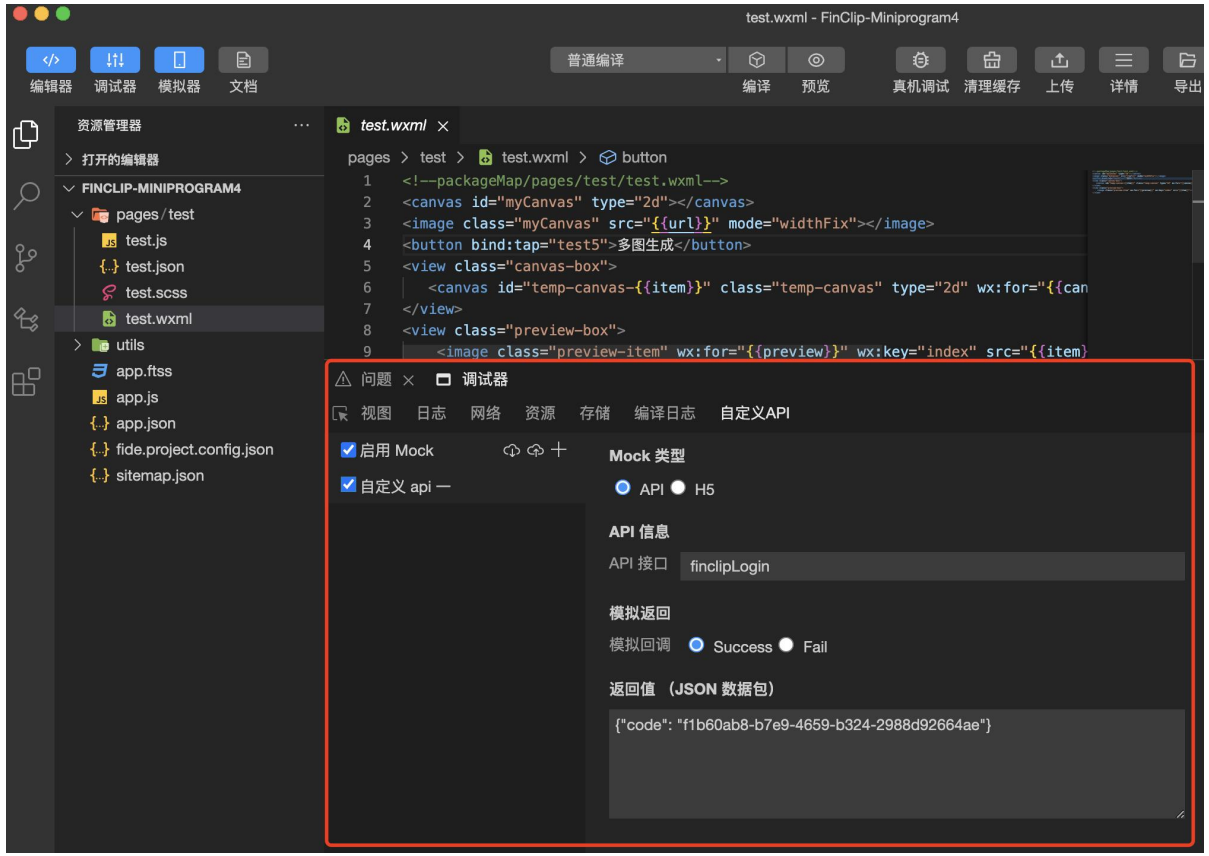
Mock自定义API

Finclip Studio 提供了 API Mock 的能力。

例如：可以通过 Mock finclipLogin 修改 ft.finclipLogin 的原生回调内容。

在小程序根目录创建 `FinClipConf.js` 并进行相应的自定义api配置

```
module.exports = {
  extApi:[
    { //普通交互API
      name: 'finclipLogin', //扩展api名 该api必须Native方实现了
      sync: false, //是否为同步api
      params: { //扩展api 的参数格式，可以只列必须的属性
        url: ''
      }
    },
    {
        name: 'finclipTestSync',
        sync: true, // 是否为同步api
        params: {
            name:'',
            title:''
        }
    }
  ]
}
```



NPM

2.3.1 安装 npm 包

在小程序 package.json 所在的目录中执行命令安装 npm 包：

```
npm install
```

此处要求参与构建 npm 的 `package.json` 需要在 `project.config.js` 定义的 `miniprogramRoot` 之内。

2.3.2 构建 npm

在小程序中使用 npm 包前，需要先构建 npm，完成 npm 构建后，会在对应的目录生成 `miniprogram_npm` 目录。

点击 FIDE 的菜单栏：工具 -> 构建 npm。

2.3.3 构建完毕后使用 npm 包

js 中引入 npm 包：

```js
const myPackage = require('packageName')
const packageOther = require('packageName/other')
```

使用 npm 包中的自定义组件（此处使用 npm 包时如果只引入包名，则默认寻找包名下的 index.js 文件或者 index 组件）：

```js
{
  "usingComponents": {
    "myPackage": "packageName",
    "package-other": "packageName/other"
  }
}
```

睿智开元

谢谢观看

上海锐图智能科技有限公司