**微前端实践**

微前端（Micro-Frontend），是将微服务（Micro-Services）理念应用于前端技术后的相关实践。

微前端的核心价值在于“技术栈无关”，那么为什么“技术栈无关”这么重要？

对于ToB 软件应用而言，软件使用寿命较长，常见也有3~5年的寿命，但是如何确保技术栈在3~5年甚至更久都生命力呢？实际上不要说跨框架，react，15 跟 16 都无法兼容的，hooks也还不能用于class component，还有软件的打包方案，目前虽然默认都采用webpack，但是webpack版本升级，也难做到每次都跟进，还有babel、less、typescript等。有几个人敢保证把自己项目一年后把所有依赖包升级到最新还能跑起来呢？

所以大部分企业应用都会有一个核心的诉求，就是如何确保我的遗产代码能平滑的迁移，以及如何确保我在若干年后还能用上时下热门的技术栈？

我认为觉得的思路应该是独立运行、独立开发、独立部署、不限技术栈。

我们只需要在主系统中构造一个足够轻量的基座及类单页面应用，然后让子应用按照共同的协议实现即可。这个协议可以包括，主应用应该如何加载子应用，以及子应用如何被主应用感知、调度、应用之间如何通信等。这个协议不应该包括，子应用要如何确保隔离性、安全性，也就是子应用除了实现一些较简单的协议之外，跟开发一个正常spa应用没有区别，包括不应该有开发、构建、发布等流程上的侵入。只要子应用实现了这几个协议，不需要关心或者干预其它内容。

**为什么要用微前端**

一般而言，“类单页面应用”实现方式有两种：

1．Iframe嵌入

2. 微前端合并单页应用

其中iframe嵌入方式比较容易实现，但是在实践过程中带来了如下问题：

* + 子项目需要改造，提供一组不带导航的功能
  + Iframe嵌入显示区大小不容易控制，存在一定局限性
  + Iframe嵌套网页URL记录完全无效，页面刷新不能够被记忆，刷新会返回首页
  + iframe的样式显示、兼容性等都具有局限性

**如何改造现有项目**

现有项目改造分为两部分一是父项目的处理，二是子项目的处理。

**概念**

什么是Single-spa？

single-spa 一个基于JavaScript的 微前端 框架，他可以用于构建可共存的微前端应用，每个前端应用都可以用自己的框架编写，完美支持 Vue React Angular。可以实现 服务注册 事件监听 子父组件通信 等功能。

主要应用于父项目集成子项目。

什么是 single-spa-vue ?

single-spa-vue 是提供给使用vue子项目使用的npm包。他可以快速和sigle-spa父项目集成，并提供了一些比较便携的api。

主要应用于子项目 使用，如果子项目使用框架为react或者angular，类似还有single-spa-react、single-spa-angular。

**父项目的改造**

1. 父项目创建dom节点，作为子项目挂载的目标dom元素。



1.1 父项目中子项目挂载目标元素

2. 注册子项目

在注册之前，我们先了解一下2个api：

singleSpa.registerApplication：这是注册子项目的方法。参数如下：

appName: 子项目名称

applicationOrLoadingFn: 子项目注册函数，用户需要返回 single-spa 的生命周期对象。

activityFn: 回调函数入参 location 对象，可以写自定义匹配路由加载规则。

singleSpa.start：这是启动函数。

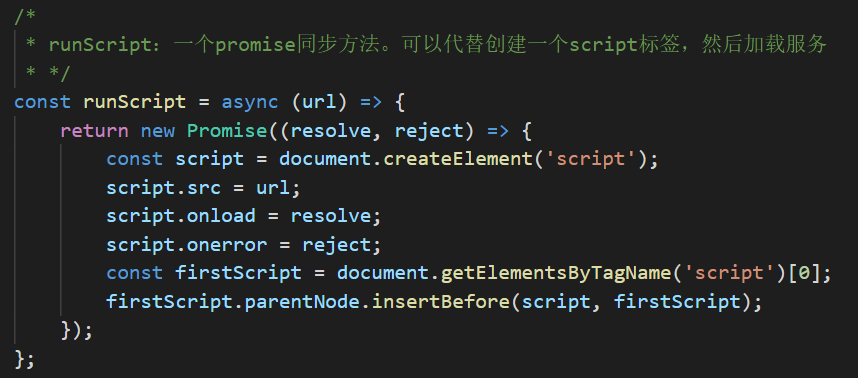


1.2 注册子项目

远程加载子项目线上js文件：

在这里，我们的远程加载使用的是async await构建一个同步执行任务。

创建一个script标签，等script加载后，返回script加载到window上面的对象。



1.3 远程加载js文件

**子项目的改造**

子项目处理比父项目稍微复杂一些，如果想注册一个子项目，需要使用single-spa-vue进行包装，因为子项目为vue框架，所以采用single-spa-vue包装。

在子项目main.js中引入single-spa-vue，传入Vue对象和App.vue挂载参数，就可以实现注册。它会返回一个对象，里面有single-spa参数需要的生命周期函数。使用export导出即可。



* 1. single-spa-vue包装子项目

子项目main.js入口文件使用single-spa-vue进行包装后会和自身Vue实例创建产生冲突，所以创建vue实例化对象时需要进行判断，如果当前环境并非在single-spa模式下，则通过new Vue进行实例化，如果在single-spa环境下则需要通过single-spa-vue进行创建。



* 1. 子项目未通过single-spa独立渲染

目前只是导出single-spa-vue包装后的子项目，还需要挂载到window中。

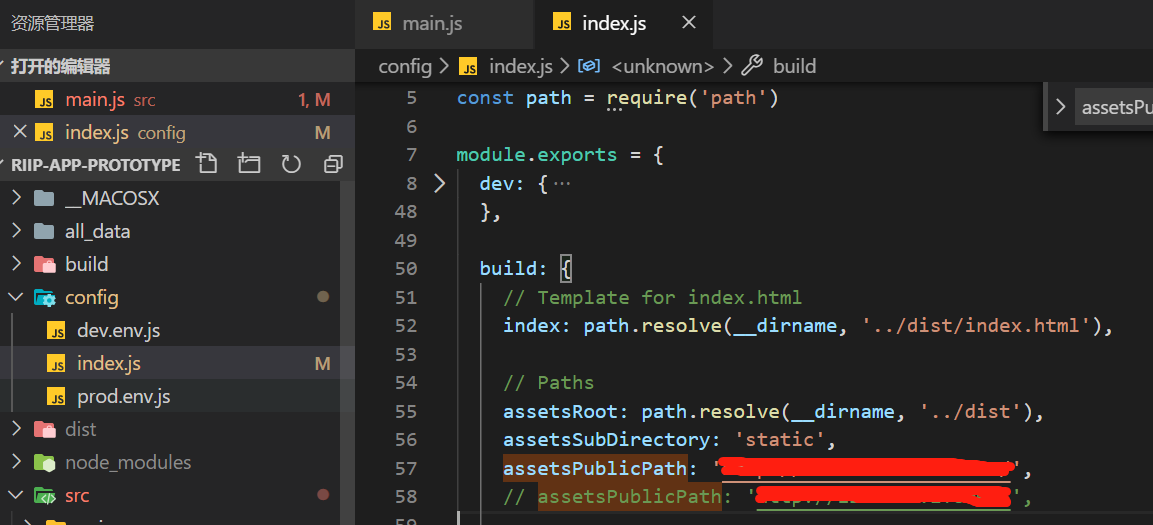
以本项目为例，在项目文件中找到build文件夹，修改我们webpack.base.conf.js文件output内的library 和 libraryTarget 字段。

* + output.library: 导出的对象名
  + output.libraryTarget: 导出后要挂载到哪里

同时，因为我们是远程调用，还需要设置 publicPath 字段为线上真实服务地址。否则父项目中加载子chunk时，会去当前浏览器域名的根路径寻找，有404问题。

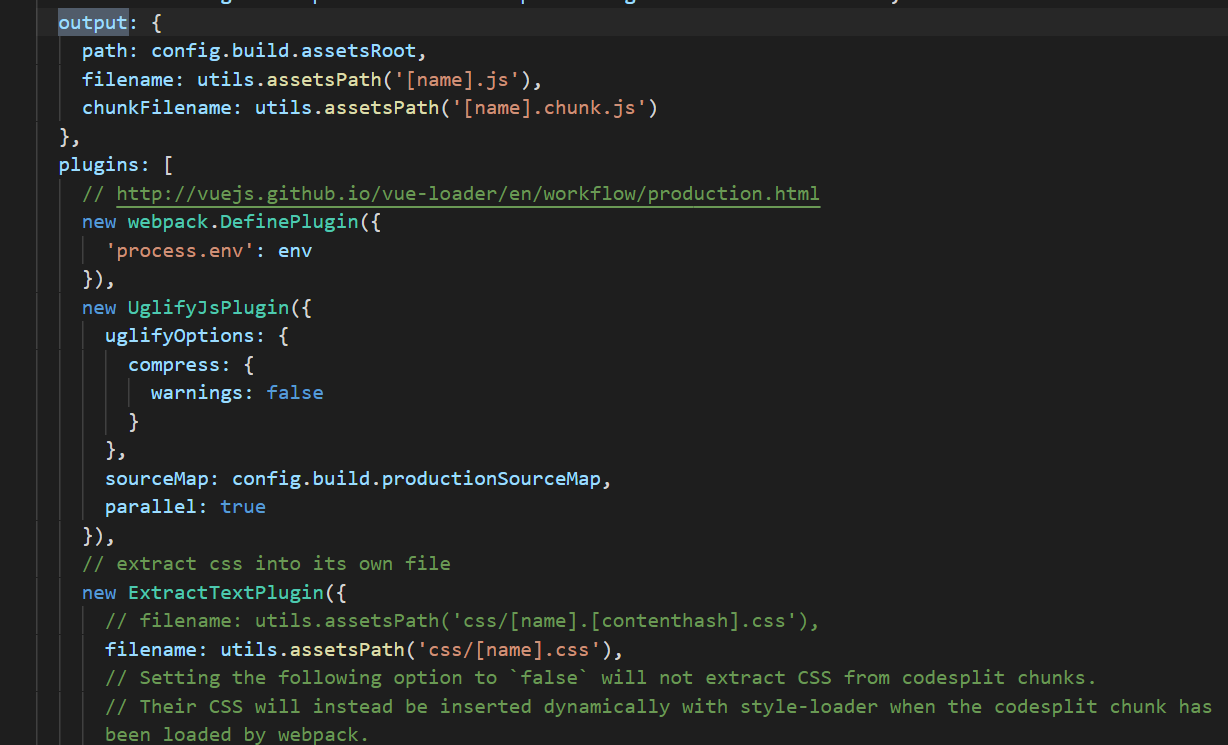


* 1. 修改出口文件配置挂载到window



* 1. 修改assetsPublicPath

为了父项目更好引入子项目打包后文件，避免每次打包线上后文件名称hash改变需要再次手动修改父项目中引入js文件名，对子项目webpack出口文件名称进行修改，取消了命名中添加的hash值。



* 1. 修改出口文件名