# 1.安装依赖

安装 vue 的官方插件 **npm i vue-class-component vue-property-decorator -S**typescript 必须安装 **npm i typescript -D**ts-loader 必须安装 **npm i ts-loader -D  
请注意：ts-loader 请与你的 webpack 版本对应，我的 webpack 版本为 3.6.0，因此我应该安装的 ts-loader 版本为 3.x.x，因此我在[ts-loader](https://links.jianshu.com/go?to=https%3A%2F%2Fgithub.com%2FTypeStrong%2Fts-loader%2Fblob%2Fmaster%2FCHANGELOG.md" \t "_blank)的[Github](https://links.jianshu.com/go?to=https%3A%2F%2Fgithub.com%2FTypeStrong%2Fts-loader%2Fblob%2Fmaster%2FCHANGELOG.md" \t "_blank)上找到了我最新的 3.x.x 的版本为 3.5.0，所以我的安装命令为**

**npm i ts-loader@3.5.0 -D**

# 2.配置 webpack.base.conf.js

首先找到./build/webpack.base.conf.js

* 找到 entry.app 将 main.js 改成 main.ts；
* 并且把项目文件中的 main.js 也改成 main.ts , 里面内容保持不变

entry: {

app: './src/main.ts'

}

* 找到resolve.extensions，里面加上.ts 后缀 （是为了之后引入.ts 的时候不写后缀）

resolve: {

extensions: ['.js', '.vue', '.json', '.ts', '.tsx'], //加入.ts

}

* 找到module.rules 添加webpack对.ts的解析

module: {

rules: [

{

test: /\.tsx?$/,

loader: 'ts-loader',

exclude: /node\_modules/,

options: {

appendTsSuffixTo: [/\.vue$/]

}

},

]

}

# 3.添加 tsconfig.json

在根路径下创建tsconfig.json文件，添加一下配置

{

*"include"*: [

        "src/\*\*/\*"

    ],

*"exclude"*: [

        "node\_modules",

        "\*\*/\*.spec.ts"

    ],

*"compilerOptions"*: {

*"allowSyntheticDefaultImports"*: true,

*"experimentalDecorators"*: true,

*"allowJs"*: true,

*"module"*: "esnext",

*"target"*: "es5", // 编译目标平台

*"moduleResolution"*: "node", //决定如何处理模块

*"isolatedModules"*: true,

*"noImplicitAny"*: true, //进行强类型检查，无法推断类型时，提示错误

*"strict"*: true, //启用所有严格类型检查选项

*"removeComments"*: true, //编译生成的js文件是否移除注释

*"lib"*: [

            "dom",

            "es5",

            "es6",

            "es7",

            "es2015.promise"

        ],

*"sourceMap"*: true, // 编译文件对应关系

*"pretty"*: true

    }

}

# 4.让 ts 识别 .vue

由于 TypeScript 默认并不支持 \*.vue后缀的文件，所以在 vue 项目中引入的时候需要创建一个 vue-shim.d.ts 文件，放在项目对应使用目录下，所以新建 src/vue-shim.d.ts，写入以下代码

declare module '\*.vue' {

import Vue from 'vue'

export default Vue

}

declare *module* 'vue/types/vue' {

*interface* Vue {

        $Message: *any*,

        $modal: *any*

    }

}

# 5..js 文件重命名为.ts 文件

将src下的所有\*\*.js文件重命名\*\*.ts，包括src/router/index.js等逐一从.js重命名为.ts

注意：重命名后对vue文件的import，需添加.vue后缀

因为Typescript默认只识别\*.ts文件，不识别\*.vue文件

之前：

import App from './App'

import HelloWorld from '@/components/HelloWorld'

需改为：

import App from './App.vue'

import HelloWorld from '@/components/HelloWorld.vue'

# 6.改造 .vue 文件

要点：

* <script>标签添加lang="ts"声明
* 使用[vue-class-component](https://links.jianshu.com/go?to=https%3A%2F%2Fgithub.com%2Fvuejs%2Fvue-class-component) 对 Vue 组件进行封装

之前：

<template>

<div id="app">

<router-view />

</div>

</template>

<script>

export default { name: 'App' }

</script>

<style>

@import './style/app.scss';

</style>

改造后：

<template>

<div id="app">

<router-view />

</div>

</template>

<script lang="ts">

import Vue from 'vue'

import Component from 'vue-class-component'

@Component

export default class App extends Vue {}

</script>

<style>

@import './style/app.scss';

</style>

之后请将所有的.vue按照[vue-class-component](https://links.jianshu.com/go?to=https%3A%2F%2Fgithub.com%2Fvuejs%2Fvue-class-component" \t "_blank)改造

# 7.运行 npm run dev

至此运行项目，即可正常运行，vue对typescript的初步引入，基本完成。

## 8.让vue识别全局方法/变量

因为在项目中，会存在自己写的一些方法是放在 vue.prototype上的。  
如：Vue.prototype.$api = myApi，然后想在每个.vue里面这样使用this.$api，然而使用ts改造之后，在.vue里面使用this.$api会报错

Property '$api' does not exist on type 'Text'

请在main.ts里面加上这段： **请放在 new Vue上面**

Vue.config.productionTip = false

// 全局方法 this. 的调用方式

declare module 'vue/types/vue' {

interface Vue {

$tool: any

$api: any

}

}

/\* eslint-disable no-new \*/

new Vue({

el: '#app',

store,

router,

components: { App },

template: '<App/>'

})

然后就能使用 this.$api 和 this.$tool了

Sass报错解决

Npm uninstall sass-loader

Npm uninstall node-sass

Npm install [sass-loader@7.3.1](mailto:sass-loader@7.3.1)

Npm install [node-sass@4.14.1](mailto:node-sass@4.14.1)

{

Test:/\.sass$/,

Loaders:[‘style’,’css’,’scss’]

}