构建基础篇 3：env 文件与环境设置

在实际项目的开发中，我们一般会经历项目的开发阶段、测试阶段和最终上线阶段，每一个阶段对于项目代码的要求可能都不尽相同，那么我们如何能够游刃有余的在不同阶段下使我们的项目呈现不同的效果，使用不同的功能呢？这里就需要引入**环境**的概念。

一般一个项目都会有以下 3 种环境：

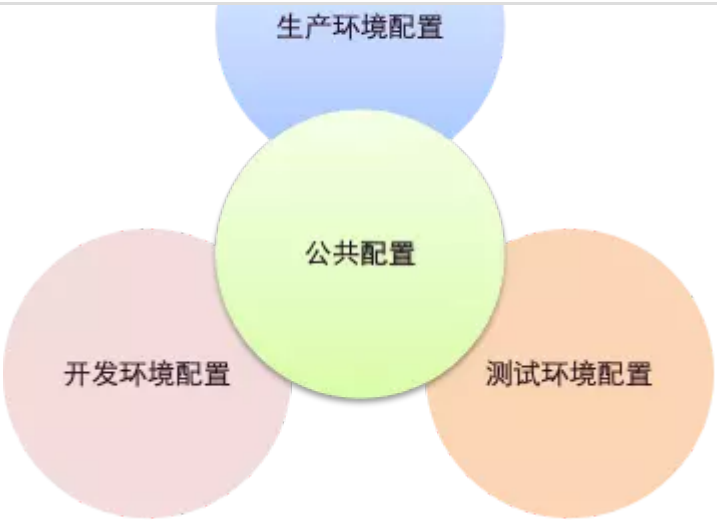
- 开发环境（开发阶段，本地开发版本，一般会使用一些调试工具或额外的辅助功能）
- 测试环境（测试阶段，上线前版本，除了一些 bug 的修复，基本不会和上线版本有很大差别）
- 生产环境（上线阶段，正式对外发布的版本，一般会进行优化，关掉错误报告）

作为一名开发人员，我们可能需要针对每一种环境编写一些不同的代码并且保证这些代码运行在正确的环境中，那么我们应该如何在代码中判断项目所处的环境同时执行不同的代码呢？这就需要我们进行正确的环境配置和管理。

介绍

1. 配置文件

正确的配置环境首先需要我们认识不同环境配置之间的关系，如图所示：



我们从上图中可以了解到每一个环境其实有其不同的配置，同时它们也存在着交集部分，交集便是它们都共有的配置项，那么在 Vue 中我们应该如何处理呢？

我们可以在根目录下创建以下形式的文件进行不同环境下变量的配置：

```
.env                # 在所有的环境中被载入
.env.local          # 在所有的环境中被载入，但会被 git 忽略
.env.[mode]         # 只在指定的模式中被载入
.env.[mode].local   # 只在指定的模式中被载入，但会被 git 忽略
```

比如我们创建一个名为 .env.stage 的文件，该文件表明其只在 stage 环境下被加载，在这个文件中，我们可以配置如下键值对的变量：

```
NODE_ENV=stage
VUE_APP_TITLE=stage mode
```

这时候我们怎么在 vue.config.js 中访问这些变量呢？很简单，使用 `process.env.[name]` 进行访问就可以了，比如：

javascript

```
// vue.config.js

console.log(process.env.NODE_ENV); // development（在终端输出）
```

令为：

```json
"scripts": {
    "serve": "vue-cli-service serve --mode stage",
}
```

`--mode stage` 其实就是修改了 webpack 4 中的 mode 配置项为 stage，同时其会读取对应 .env.[model] 文件下的配置，如果没找到对应配置文件，其会使用默认环境 development，同样 `vue-cli-service build` 会使用默认环境 production。

这时候如果你再创建一个 .env 的文件，再次配置重复的变量，但是值不同，如：

```
NODE_ENV=staging
VUE_APP_TITLE=staging mode
VUE_APP_NAME=project
```

因为 .env 文件会被所有环境加载，即公共配置，那么最终我们运行 `vue-cli-service serve` 打印出来的是哪个呢？答案是 **stage**，但是如果是 .env.stage.local 文件中配置成上方这样，答案便是 **staging**，所以 .env.[mode].local 会覆盖 .env.[mode] 下的相同配置。同理 .env.local 会覆盖 .env 下的相同配置。

由此可以得出结论，相同配置项的权重：

```
.env.[mode].local > .env.[mode] > .env.local > .env
```

但是需要注意的是，除了相同配置项权重大的覆盖小的，不同配置项它们会进行合并操作，类似于 Javascript 中的 Object.assign 的用法。

2. 环境注入

通过上述配置文件的创建，我们成功使用命令行的形式对项目环境进行了设置并可以自由切换，但是需要注意的是我们在 Vue 的前端代码中打印出的 `process.env` 与 vue.config.js 中输出的可能是不一样的，这需要普及一个知识点：webpack 通过 DefinePlugin 内置插件将 process.env 注入到客户端代码中。

```
    ...

    plugins: [
        new webpack.DefinePlugin({
            'process.env': {
                NODE_ENV: JSON.stringify(process.env.NODE_ENV)
            }
        }),
    ],

    ...
}
```

由于 vue-cli 3.x 封装的 webpack 配置中已经帮我们完成了这个功能，所以我们可以直接在客户端代码中打印出 process.env 的值，该对象可以包含多个键值对，也就是说可以注入多个值，但是经过 CLI 封装后仅支持注入环境配置文件中以 `VUE_APP_` 开头的变量，而 `NODE_ENV` 和 `BASE_URL` 这两个特殊变量除外。比如我们在权重最高的 .env.stage.local 文件中写入：

```
NODE_ENV=stage2
VUE_APP_TITLE=stage mode2
NAME=vue
```

然后我们尝试在 vue.config.js 中打印 `process.env`，终端输出：

```json
{
    ...

    npm_config_ignore_scripts: '',
    npm_config_version_git_sign: '',
    npm_config_ignore_optional: '',
    npm_config_init_version: '1.0.0',
    npm_package_dependencies_vue_router: '^3.0.1',
    npm_config_version_tag_prefix: 'v',
    npm_node_execpath: '/usr/local/bin/node',
    NODE_ENV: 'stage2',
    VUE_APP_TITLE: 'stage mode2',
    NAME: 'vue',
    BABEL_ENV: 'development',
```

可以看到输出内容除了我们环境配置中的变量外还包含了很多 npm 的信息，但是我们在入口文件 main.js 中打印会发现输出：

```json
{
    "BASE_URL": "/vue/",
    "NODE_ENV": "stage2",
    "VUE_APP_TITLE": "stage mode2"
}
```

可见注入时过滤调了非 `VUE_APP_` 开头的变量，其中多出的 `BASE_URL` 为你在 vue.config.js 设置的值，默认为 /，其在环境配置文件中设置无效。

3. 额外配置

以上我们通过新建配置文件的方式为项目不同环境配置不同的变量值，能够实现项目基本的环境管理，但是 .env 这样的配置文件中的参数目前只支持静态值，无法使用动态参数，在某些情况下无法实现特定需求，这时候我们可以在根目录下新建 config 文件夹用于存放一些额外的配置文件。

```
// 公共变量
const com = {
    IP: JSON.stringify('xxx')
};

module.exports = {

    // 开发环境变量
    dev: {
        env: {
            TYPE: JSON.stringify('dev'),
            ...com
        }
    },

    // 生产环境变量
    build: {
        env: {
            TYPE: JSON.stringify('prod'),
            ...com
        }
    }
}
```

上方代码我们把环境变量分为了公共变量、开发环境变量和生产环境变量，当然这些变量可能是动态的，比如用户的 ip 等。现在我们要在 vue.config.js 里注入这些变量，我们可以使用 chainWebpack 修改 DefinePlugin 中的值：

javascript

```
/* vue.config.js */
const configs = require('./config');

// 用于做相应的 merge 处理
const merge = require('webpack-merge');

// 根据环境判断使用哪份配置
const cfg = process.env.NODE_ENV === 'production' ? configs.build.env : configs.dev.

module.exports = {
    ...

    chainWebpack: config => {
```

```
                // 使用 merge 保证原始值不变
                args[0][name] = merge(args[0][name], cfg);

                return args
            })
    },

    ...
}
```

最后我们可以在客户端成功打印出包含动态配置的对象：

```json
{
    "NODE_ENV": "stage2",
    "VUE_APP_TITLE": "stage mode2",
    "BASE_URL": "/vue/",
    "TYPE": "dev",
    "IP": "xxx"
}
```

4. 实际场景

结合以上环境变量的配置，我们项目中一般会遇到一些实际场景： 比如在非线上环境我们可以给自己的移动端项目开启 vConsole 调试，但是在线上环境肯定不需要开启这一功能，我们可以在入口文件中进行设置，代码如下：

```javascript
/* main.js */

import Vue from 'vue'
import App from './App.vue'
import router from './router'
import store from './store'

Vue.config.productionTip = false

// 如果是非正式环境，加载 VConsole
if (process.env.NODE_ENV !== 'production') {
```

```
new Vue({
  router,
  store,
  render: h => h(App)
}).$mount('#app')
```

vConsole 是一款用于移动网页的轻量级，可扩展的前端开发工具，可以看作是移动端浏览器的控制台，如图：

另外我们还可以使用配置中的 BASE_URL 来设置路由的 base 参数：

```
import Vue from 'vue'
import Router from 'vue-router'
import Home from './views/Home.vue'
import About from './views/About.vue'

Vue.use(Router)

let base = `${process.env.BASE_URL}`; // 获取二级目录

export default new Router({
    mode: 'history',
    base: base, // 设置 base 值
    routes: [
        {
            path: '/',
            name: 'home',
            component: Home
        },
        {
            path: '/about',
            name: 'about',
            component: About
        }
    ]
})
```

每一个环境变量你都可以用于项目的一些地方，它提供给了我们一种全局的可访问形式，也是基于 Node 开发的特性所在。

结语

环境的配置和管理对于项目的构建起到了至关重要的作用，通过给项目配置不同的环境不仅可以增加开发的灵活性、提高程序的拓展性，同时也有助于帮助我们去了解并分析项目在不同环境下的运行机制，建立全局观念。

思考 & 作业

- `process.env` 中如何获取 package.json 中 name 的值?
- 如何在 package.json 中的 scripts 字段中定义一些自定义脚本来切换不同的环境?

留言

评论将在后台进行审核，审核通过后对所有人可见

qiqingfu

process.env.NODE_ENV 为什么要进行 JSON.stringify 处理，是因为webpack内部处理的时候把它作为一个变量了吗?类似与eval('production'),所以要JSON. stringify()转成字符串

▲ 0　收起评论　1月前

小甘子

我认为变成字符窜基本类型可以实现深度拷贝

1月前

评论审核通过后显示　评论

sonicsunsky

思考&作业没答案吗?

▲ 0　评论　1月前

aierong dg @ dg

没看懂第3点：额外配置。怎么实现动态的参数了？请教一下！

我把公用的配置在.evn文件,其它的配置在.env.production，或者.env.development ，好像是一样的达到效果

爱烤火的鱼

有没有一套完整的demo或教程

▲ 0　　评论　2月前

砸了我的锅的铁 web 前端工程师

我来回答第一个。因为环境变量在.env里面的配置时ini格式的，而在取值进行比较的时候，需要进行字符串转换，这个process.env.NODE_ENV === 'production'才能是true

▲ 0　　评论　2月前

少说话多做事hg

作业哪里有答案看呢?

▲ 0　　收起评论　2月前

劳卜　前端工程师 @ TC

暂时没有固定答案，可以参考评论自己思考下，再尝试百度比较好

2月前

评论审核通过后显示　　评论

少说话多做事hg

我想问下，build出来放在线上，还能访问这些环境变量吗

▲ 0　　收起评论　2月前

劳卜　前端工程师 @ TC

可以的

2月前

评论审核通过后显示　　评论

start.prod : cross-env NODE_ENV=production node server/index.js
}
这样对吗

▲ 0　　评论　2月前

Bug输出小能手 前端工程师
process.env.npm_package_name

▲ 0　　评论　2月前

黄天马君
// vue.config.js

console.log(process.env.NODE_ENV); // development（在终端输出）

这里应该是MODE_ENV

▲ 1　　收起评论　2月前

黄天马君
我搞错了，都是NODE_ENV
2月前

评论审核通过后显示　　评论

Honter
完全看不懂啊😭

▲ 0　　收起评论　2月前

tenyiyi 前端工程师 @ 小码农
me too不懂这一节到底用来干啥
2月前

NorthSeacoder

额外配置的com中间的IP: JSON.stringify后面的反括号是中文括号。。。。。

▲ 0　收起评论　2月前

> **劳卜** 前端工程师 @ TC
> 火眼金睛，已修正。。
> 2月前
>
> 评论审核通过后显示　评论

木子烁束岸 前端端端端端端端端端端端端端端端端...

这两章好抽象。。。

▲ 0　评论　2月前

RonaldZhao 全干工程师养成ing

看完这两节感觉什么都没学会？是因为我不是搞前端的吗？

▲ 0　评论　2月前

Hector本尊 前端开发

在git上的例子里vue.config.js文件中第10行const isPro = process.env.NODE_ENV === 'production'，这里无论怎么打包process.env.NODE_ENV始终是undefined?

▲ 0　收起评论　2月前

> **Hector本尊** 前端开发
> 我看错了，在package.json中少了"serve": "vue-cli-service serve --mode dev",
> 2月前
>
> 评论审核通过后显示　评论

▲ 0　　评论　2月前

Si

2. 在 config/index.js 中引入 package, 然后再将其 name 属性加到 process.env 的变量中
1. 为什么要用 JSON.stringify, 第二题的 name 若有 –, 就会报错(如: my–project), JSON.stringify() 可以确保命名为字符串
3. 应该是通过 ––mode envName 来切换环境

▲ 6　　评论　3月前

Chuck1478262462000 前端开发 @ 创客贴

在 node 端，x = 'test'，会将代码里的 x 变量全部替换为 test；而 x = '"test"'，则会将代码里的 x 变量全部替换为"test"

▲ 0　　评论　3月前

林振杰 前端开发工程师

.env能存放对象吗？比方说我和后端是根据命令号来路由的，环境变量 API: {
  'login': 100001,
  'logout': 100002
}

前段js代码写是 http.post(API.logout), 但编译后是http.post(100002)，在vue cli3要怎么实现?

▲ 0　　收起评论　3月前

劳卜 前端工程师 @ TC

不行，可以看本章的“额外配置”部分

3月前

aierong dg @ dg

好像不可以存对象吧！一个个定义吧

1月前

评论审核通过后显示　　评论

thisthis 全栈工程师

看完vue封装的配置 感觉是弊大于利

▲ 1　收起评论　3月前

青色

越懂越不懂

3月前　回复

Murphy君

同感，本来webpack4文档挺清楚的 一封装 不知道怎么配

3月前

评论审核通过后显示　评论

江湖人称李老板 前端工程师 @ 英泰

在公司上班的时候看完了这一章,马上就解决了我项目中遇到的问题,很6

▲ 0　收起评论　3月前

劳卜 前端工程师 @ TC

厉害了

3月前

毛绒玩具皮卡丘01

前端搬砖，努力成为一个超级水的full-stack engineer鸭 @ 搬砖搬砖

写angular，但项目中基本没有去配置环境，心很慌

3月前

Hector本尊 前端开发

不知道有没有修改配置文件如config.js，不需要重新打包，这种动态获取配置的方法。

2月前

评论审核通过后显示　评论