【yeahmobi】新前端框架NFF项目  
详细设计

|  |  |
| --- | --- |
| 项目名称 | 新前端框架NFF详细设计 |
| 项目负责人 | 杨斌 |
| 文档作者 | 杨斌，马爱莲 |
| 提交日期 | 2017.2.22 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 版本号 | 修改内容 | 日期 | 修改人 |
| 1 | 0.1.0 | 创建 | 2017.2.22 | 杨斌 |
| 2 | 0.1.1 | Review文档 | 2017.2.28 | 杨斌 |
|  |  |  |  |  |

目录

[【yeahmobi】新前端框架NFF项目 详细设计 1](#_Toc477339770)

[1 背景 4](#_Toc477339771)

[1.1 现状 4](#_Toc477339772)

[1.2 名词解释 4](#_Toc477339773)

[2 目标 5](#_Toc477339774)

[3 系统环境 5](#_Toc477339775)

[3.1 操作系统以及语言环境 5](#_Toc477339776)

[3.2 数据量及规模 6](#_Toc477339777)

[4 设计思路与折中 6](#_Toc477339778)

[5 详细设计 7](#_Toc477339779)

[5.1 总体架构图 7](#_Toc477339780)

[5.2 用例分析—aff端登录页面 8](#_Toc477339781)

[6 落地与实施 8](#_Toc477339782)

[6.1 规划 8](#_Toc477339783)

[6.2 开发工时评估 8](#_Toc477339784)

[6.3 开发难点风险 9](#_Toc477339785)

[7 参考资料 9](#_Toc477339786)

# 背景

现有系统前端框架难以承担新的移动端开发需求，并且前端技术栈过于陈旧，依赖许多第三方库和框架，已很难维护升级，在性能以及开发效率方便都存在瓶颈，并缺乏必要的前端工程化工具。故此，为提升前端技术栈，面对新的移动端开发需求，基于vuejs开发设计出新的前端开发框架NFF

## 现状

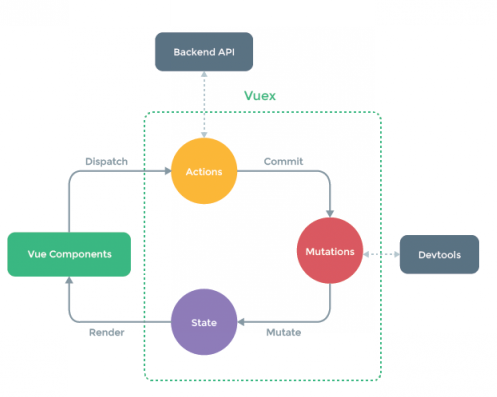
目前现有的开发框架是基于项目启动时期的相对简单的需求，只做了简单的框架分层，并利用了大量的第三方框架，目前已十分庞大，难以应对目前移动化时代app不同终端的需求，开发效率存在瓶颈。

## 名词解释

1. 前端工程：为了快速开发前端应用而开发的开发集成工具。提供前端开发调试、代码打包部署、单元测试，开发数据mock等。

2. Vuejs：目前十分流行的前端MVVM框架，不同与传统的dom编程模式，vuejs利用的是数据驱动应用的思想，即组件pros和state来管理数据，并控制页面的渲染更新，不同与react框架的model引发view 更新机制，实现原理是利用es6中提供的Object.defineProperty支持数据状态响应性，最新版2.0提供了大量的语法糖设计，可以很迅速的写出优雅和高效的代码。并且基于webpack构建，可以十分便捷的开发与测试。

3. Vuex：vuejs提出的状态管理规范，利用全局单例模式store管理程序的状态。并严格定义各概念之间的约束和规则，使代码变得高度维护和结构化，非常适用于不同组件共享同一状态等应用场景，与redux类似。（如下图）



4. Es6：最新一代的javascript语言规范，提供了promise异步编程，CommonJS模块，对象扩展符等非常实用的api，增强了前端技术的表现力和模块化

5. 组件化：根据关注点分离设计思想，将view渲染，数据状态管理，后台数据获取，权限校验等切面处理等分离，各模块之间保持细颗粒度。

# 目标

基于vuejs开发出灵活、高效、高可维护性、高性能的开发框架，并负责新框架的落地实施，开发出移动端联盟系统（手机浏览器触屏版），重构现有联盟系统前端框架，统一现有的前端技术框架，提高前端技术栈。

# 系统环境

## 操作系统以及语言环境

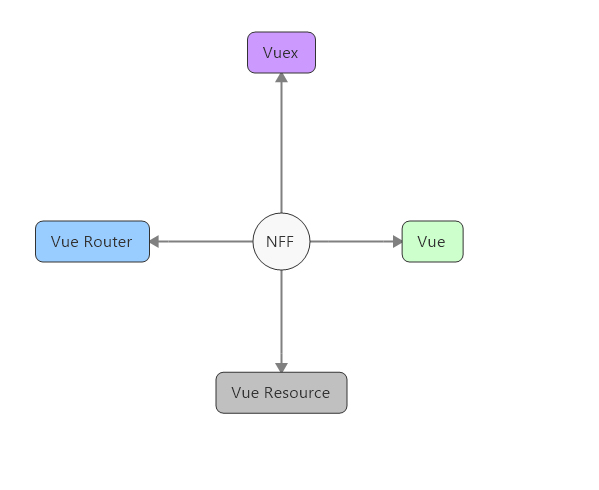
联盟前端技术栈

## 数据量及规模

现有的联盟桌面端aff和manage系统，以及计划中的移动触屏端联盟系统。

# 设计思路与折中

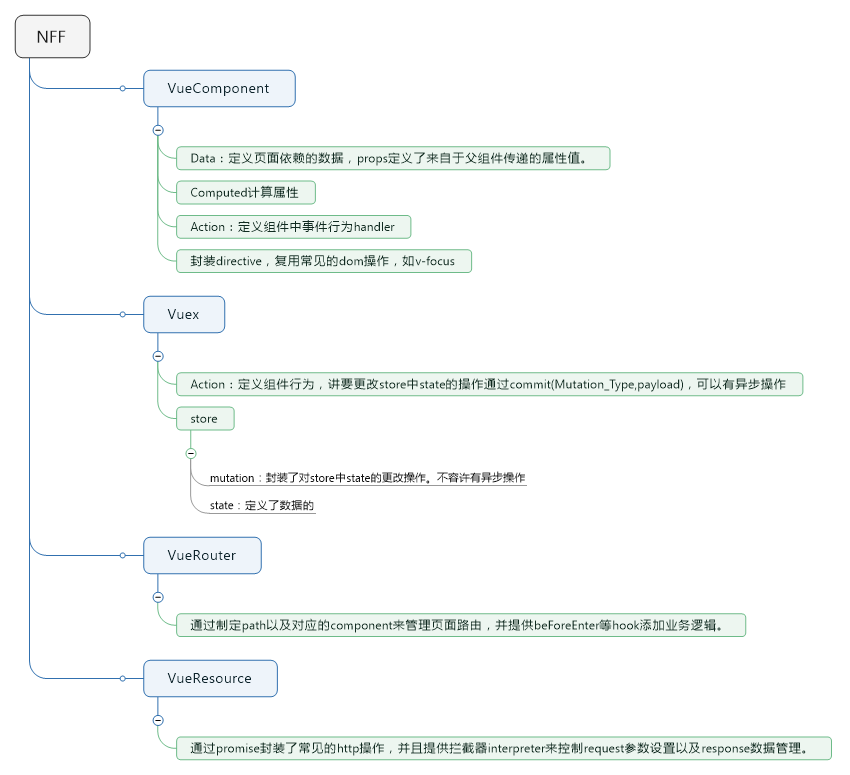
NFF技术栈包括：VueJS+VueRouter+Vuex+VueResource，其中Vuejs负责组件定义，VueRouter负责页面路由定义控制，Vuex负责页面数据以及数据状态管理，VueResource负责封装http请求服务。框架图如：



开发调试：利用vue官方的vue-tools chrome extension，开发利用webpack-dev-server启动vue组件，支持热加载。

# 详细设计

## 总体架构图



Vue定义的组件分为<template><script><style>

1.其中template采取定义页面模块，利用vue提供的指令规范定义页面数据和行为。如<span>{{comp.name}}</span>表示此处填充模块定义的数据comp属性值name;<button @click=”clickBtn”>A Button</button>表示点击button后的handler为clickBtn,在组件里面定义。

2.script定义模块中依赖的数据以及事件触发后的method。

3.style定义页面样式。

Vue组件声明定义了以下规范：

1.data定义简单的数据，供模板中初始时使用。

2.computed计算属性，定义了基于data属性值简单运算后的值。默认会缓存起来。有助于提升性能。

3.action定义了模板中输出的行为。

Vue指令是封装了常见的行为操作，如v-focus封装了对表单控件的自动聚焦。

Vue过滤器定义了对数据的过滤转换操作。如对页面显示字段的本地化转换。

## 用例分析—aff端登录页面及项目文档

详见：<http://172.30.10.66:8981/ui_library> <http://git.dy/yeahmobi/ymmanage/issues>

# 落地与实施

## 规划

利用手机浏览器触屏端aff系统开发机会，落地实施此新前端框架NFF

## 开发工时评估

考虑到此次为初次使用新前端框架，尚有许多大量基础性技术问题如利用vuejs组件规范封装现有零散的UI组件，如何分解简化现有复杂页面，完善开发脚手架等。预计开发触屏版aff需时1个月（不包含测试，需求等工作）。以重构现有affiliate端为例，涉及开发页面及工时：

登录和忘记密码页面和注册页面 工时10h

左边栏导航和顶部导航 工时 8h

Offer列表页面以及查询组件 工时 5h

Smart link以及离线apk 工时 4h

Myprofile以及修改密码等页面 工时 10h

Offer详情页面以及相似offer模块 工时 12h

上传文件页面 工时 8h

Billing模块 工时 6h

普通报表 工时15h

转化报表 工时 8h

佣金报表 工时 4h

cps报表 工时 4h

cps转化报表 工时 4h

publisher模块页面 工时20h

共计106h，以40%工时考虑，每天工时3.5小时，需30天工作日。触屏端考虑到页面相对有所简化，但交互、体验以及性能方面会有更高要求，工时应该持平。

## 开发难点风险

1.开发思路需要调整，vuejs提供的以组件封装为核心的开发思路，是与此前jQuery提倡的dom为核心的开发最大的不同.

2.vuejs所宣称的高性能的响应性设计，优雅的编程习惯，是否在项目实施中能够验证，尤其当项目变得越来越庞大。

3.vuejs的文档相对比较完善，开发编程习惯比较灵活，但也存在中文文档比较欠缺，许多知识概念叙述比较抽象，缺乏足够demo和教程，需要一定的学习成本。

4.目前尚无详尽的开发基础教程和脚手架等开发帮手，对于其他开发同事，存在一定开发门槛。

# 参考资料

1. <https://router.vuejs.org/en/>

2. <https://vuejs.org/v2/guide/>

3. <https://vuex.vuejs.org/en/>

4. <https://github.com/pagekit/vue-resource>