基于SAM模式的VueJS渐进式开发

【**摘要**】SAM是一种新的函数响应式模式，它通过严格地将业务逻辑与视图分离来简化前端架构。SAM建议将图形用户界面底层的业务逻辑分为三个概念：操作（Action）、模型（Model）和状态（State）：操作向模型供值，仅模型可以接受这样的值，模型接受这个值，通过状态来呈现视图。

VueJS 是一套基于SAM模式的构建用户界面的 渐进式框架。目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。

本文的应用场景是，是上海公安大数据汇聚平台系统中，在前后端分离的技术模型下，采用VueJs2.0渐进式框架，实现前端UI模板式开发。

【关键字】SAM VueJS Webpack Bootstrap Npm

一 **前言**

为了有效保证和响应拓展项目《北京联通沃安心项目》 和 《上海智慧公安物联网数据汇聚系统》的快速开发和快速迭代的要求，提升用户对公司技术实力的认可，降低传统java web开发项目中，视图层和Java数据控制层/模型层的强关联，严重耦合的劣性，故采用先进的实现了SAM模式的VueJS框架进行UI侧view层面的功能开发。

二 **正文**

1. 可行性分析

对SAM模式及vuejs进行可行性研究分析，如下：

**领域发展要求**：在不断发展的JavaScript编程领域，响应式编程技术正变得风生水起。

**技术成熟度**：SAM模式或者MVVM模式，以及VueJS框架，目前社区活跃，阿里、腾讯等互联网巨头的许多项目都在使用。版本更新不断，不断提供更优越性能和API。

**学习曲线**：SAM思想专注于恰当地实现“应用程序状态变化”。VueJS会让状态管理变得非常简单且直观。VueJS的API是公认的非常易于上手的GUIDE。与其他重量级框架不同的是，Vue 采用自底向上增量开发的设计。Vue 的核心库只关注视图层，并且非常容易学习，非常容易与其它库或已有项目进行整合。

**框架集成度：**VueJS非常适合单页面应用的开发，适用于前后端分离的功能开发场景，只需要JAVA REST接口暴露出来一个JSON（或者其他约定格式）对象来提供具体业务内容，即可完成功能的对接，实现零耦合。

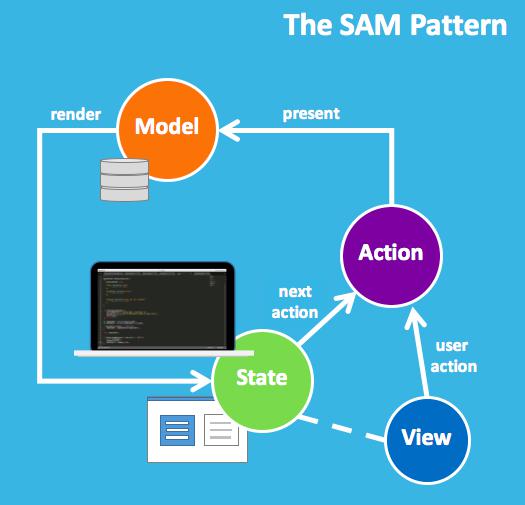
1. 技术介绍

**SAM思想剖析：**

1. SAM将图形用户界面底层的业务逻辑分为三个概念： 操作（Action）、模型（Model）和状态（State）：操作向模型供值，仅模型可以接受这样的值，模型接受这个值，通过状态来呈现视图。
2. SAM符合[软件架构的原则](https://msdn.microsoft.com/en-us/library/ee658124.aspx)：

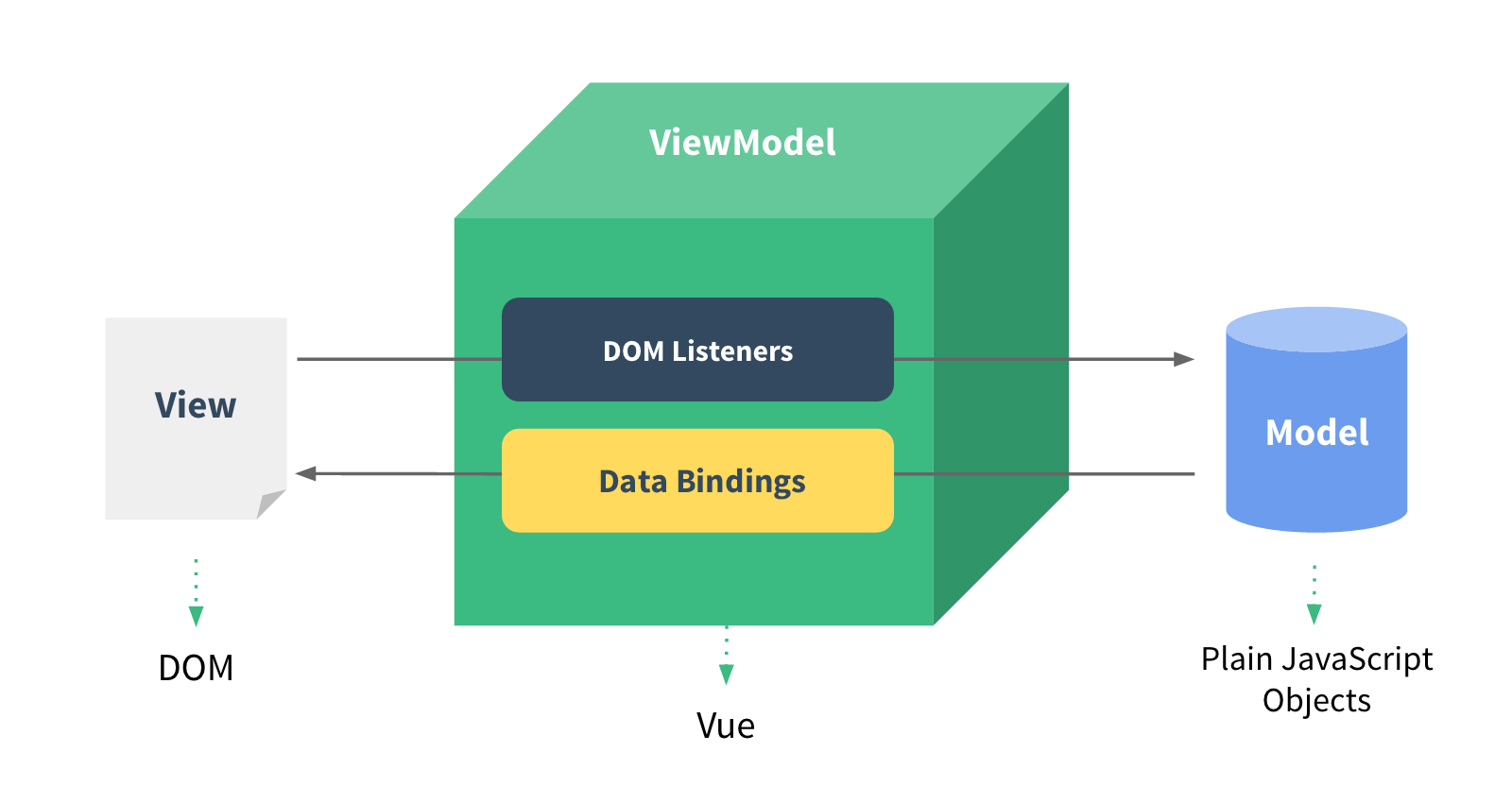
* 关注点分离：将应用程序拆分为不同的功能，并确保不同部分在功能上的重叠范围尽可能小。
* 单一职责：每个组件或模块只应承担某一特定功能或特性，或内聚功能的聚合。
* 最少知识：一个组件或对象不应了解其他组件或对象的内部细节。
* 不要重复自己：只需要在一个位置指定意图。例如在应用程序设计过程中，一个具体的功能只要在一个组件中实现即可，该功能不应重复出现在任何其他组件中。

1. SAM模式内各部分组件流程可以通过下图来展示：



**VueJS框架介绍：**

1. [Vue.js](https://github.com/vuejs/vue)是当下很火的一个SAM库，它是以数据驱动和组件化的思想构建的。VueJS能够实现SAM所提到的状态绑定操作，即使用状态将view和model相互影响，实现双向绑定，如下图所示：



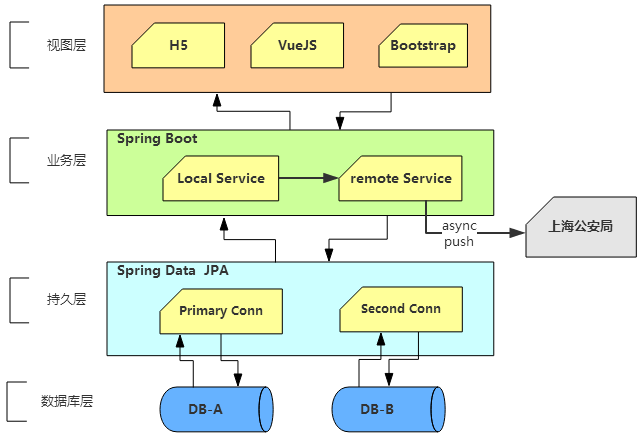
1. **VueJS跟传统View实现方式的对比**

传统java web开发项目中的view层，比如jsp方式或者jsf方式，和java数据控制层/模型层是强关联的，严重耦合的。导致美工对视图层做的修改，UI开发人员必须更新集成视图和对应的组件，每个UI功能的调整都是“牵一发而动全身”。而且美工和开发人员技术栈不同，交互过程产生的时间消耗，严重影响功能上线时间。对于新拓展项目快速开发、快速迭代的要求，弊端过于明显。

使用SAM模式替换MVC模式，视图层和模型层完全隔离，后台数据通过rest接口暴露json格式的数据。前台使用VueJs异步获取到来自后台rest接口的json数据，然后把这些数据转化为内部的虚拟对象，并且去填充自定义的组件，实现数据的双向绑定。基于此SAM模式，前台视图层有自己的一套虚拟对象环境，和后台的数据来源层是完全隔离的。美工人员使用自己的技术栈直接调整视图层，减少了和界面开发人员更新集成视图的交互工作；另外，每个UI功能的调整，只需对应调整部分组件内容即可，组件的责任链更加清晰，开发更加高效，有力保障了功能的快速上线。

1. **VueJS在系统架构中的位置及工作流程**

在<上海智慧公安物联网数据汇聚系统>项目中，整体采用了如下的架构：



系统整体采用SpringBoot+Spring Data Jpa+VueJS的基本架构。其中业务层、持久层等部分在当前文章中不再赘述，单讲VueJS部分。

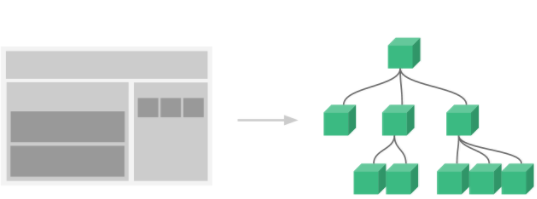
从图中可以看到，VueJS位于视图层，取代了传统的JSP等视图架构的位置。VueJS将请求通过ajax方式按照指定url发送给后端的REST接口，REST业务接口是无状态的，接收到参数后，执行对应的业务方法，调用所属的JPA Repository去操作数据表映射到的实体对象。最终生成VueJS希望接收的JSON对象字符串，返回给UI视图层去呈现想要的界面效果。

1. VueJS的特性及详细体现
2. 生命周期

每个 Vue 实例在被创建之前都要经过一系列的初始化过程。实例需要配置数据观测、编译模版、挂载实例到 DOM，然后在数据变化时更新 DOM。下图说明了实例的生命周期：



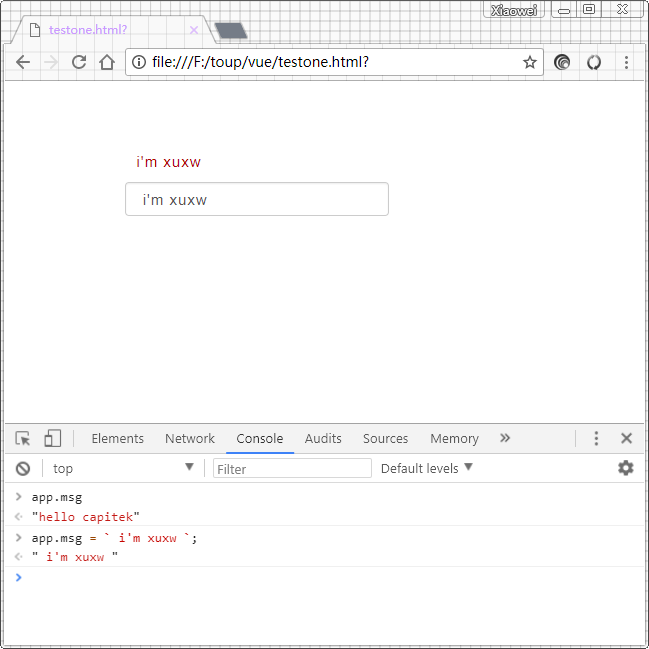
1. VueJS具备的一些特性
2. 声明式渲染：允许你采用简洁的模板语法来声明式的将数据渲染进 DOM 的系统。
3. 构建组件： 组件系统是 Vue.js 另一个重要概念， 因为它提供了一种抽象，让我们可以用独立可复用的小组件来构建大型应用，几乎任意类型的应用的界面都可以抽象为一个组件树。



1. 计算属性。为了让模板在绑定表达式比较复杂的情况下依然可以简单和清晰的处理逻辑，可以使用计算属性。
2. 事件处理器。
3. VueJS的一个数据双向绑定的简单实现

|  |
| --- |
| <div id="app">  <p>{{ message }}</p>  <input v-model="message"></input>  </div>  <script>  var app7 = new Vue({  el:'#app',  data:{  message:'hello world!'  }  });  </script> |

**最终演示的静态效果如下图所示：**



说明：文本框中输入的内容会被保存成vue的state对象，或者view-model对象内容msg，可以app.msg执行setter或者getter操作。

**演示的动态效果可以查看这个gif动画所示：**



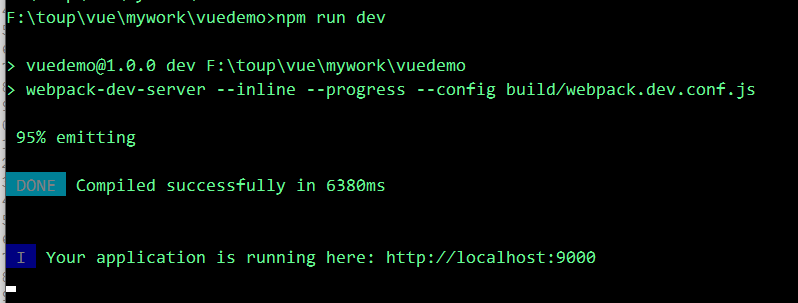
1. 上边只是一个最简单的双向数据绑定的例子。在实际的项目应用中，直接使用npm的webpack模板来创建vue实例。

首先是安装nodejs环境，同时npm也被安装。

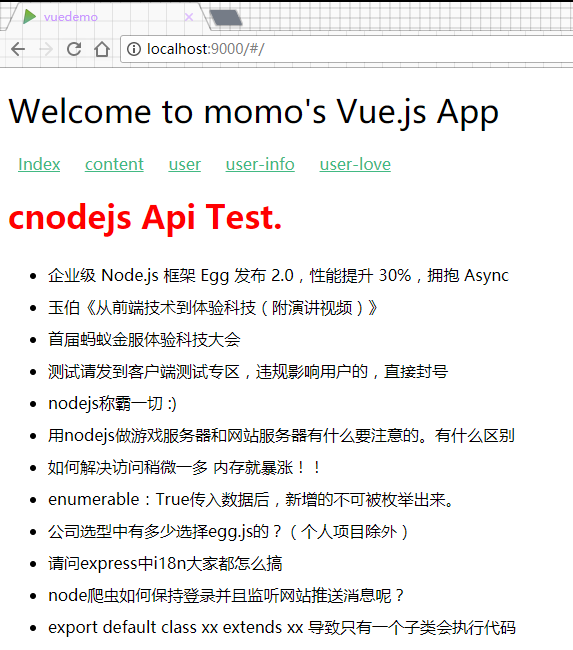
之后通过npm安装vue-cli工具，它是vuejs提供的一个脚手架工具，基于vue-cli，可以直接创建vuejs开发的全家桶工具集，例如包括vuejs-core，vuejs-route，以及需要对js，css，html等不同模板进行编译的插件，如babel插件、sass插件，html插件等等。Vue-cli，一步搞定。

最后基于npm进行启动服务：

命令行输入：npm run dev



可以直接访问vue-webpack模板工程，如下图：



**三 结束语**

通过使用渐进式的开发框架VueJS，与view层彻底跟后端控制层和业务数据层进行隔离，将传统的view层按照SAM模式分离为状态，操作和实体，实现数据的双向绑定功能，根据需求定制各类页面模板， 将不同组件的数据通过vuejs的虚拟dom进行托管，使得上下文能够管理vue对象及属性、方法、过滤器、变量监听等等，通过异步方式获取远程rest接口数据，降低前后台的代码耦合度，保证项目需求的快速响应和代码实现的更新，加快项目的迭代演进。

**四 文章讨论内容点涉及到的技术参考网站：**

Sam模式网站： <http://sam.js.org/>

VueJS官网及Guide：<http://doc.vue-js.com/v2/guide/>

Nodejs中文社区：<https://cnodejs.org/>

Vuejs-templates Github：<https://github.com/vuejs-templates>