

JOB

伯乐

BOLE

在线

马哥教育

www.magedu.com

2016全新Linux运维实战+python实战班震撼上线!

先就业,后付款!年薪24W!问徒班还有少量名额!

首页

最新文章

开发

IT技术

职场

业界

极客

创业

访谈

在国外

- 导航条 -

伯乐在线 > 首页 > 所有文章 > Web前端 > 前后端分离的思考与实践（五）

分享到:

4

神奇的JpGraph类库

WEB在线文件管理器

首页

资讯

文章

频道

资源

小组

相亲

频道

登录

注册

原文出处: [淘宝UED - 筱谷](#)

## 基于前后端分离的多终端适配

### 前言

近年来各站点基于 Web 的多终端适配进行得如火如荼，行业间也发展出依赖各种技术的解决方案。有如基于浏览器原生 CSS3 Media Query 的响应式设计、基于云端智能重排的「云适配」方案等。本文则主要探讨在前后端分离基础下的多终端适配方案。

#### 关于前后端分离

关于前后端分离的方案，在《[前后端分离的思考与实践（一）](#)》中有非常清晰的解释。我们在服务端接口和浏览器之间引入 NodeJS 作为渲染层，因为 NodeJS 层彻底与数据抽离，同时无需关心大量的业务逻辑，所以十分适合在这一层进行多终端的适配工作。

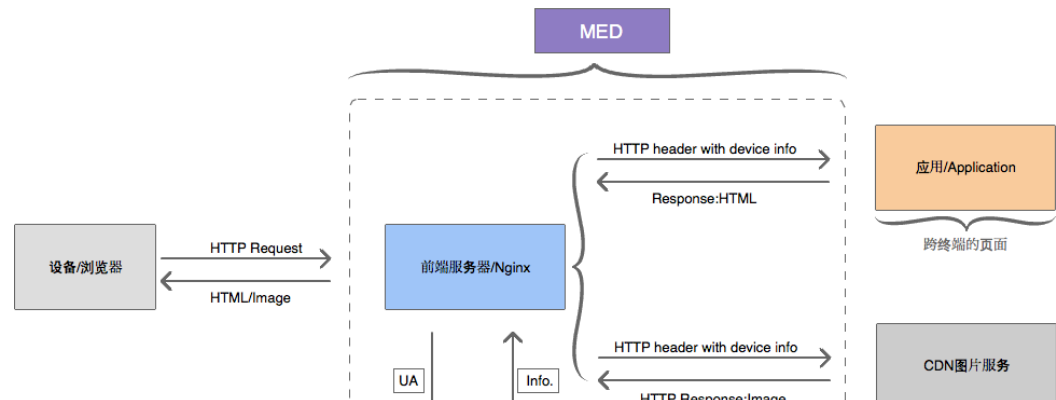
### UA 探测

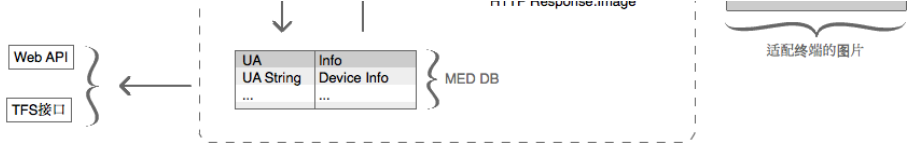
进行多终端适配首先要解决的是 UA 探测问题，对于一个过来的请求，我们需要知道这个设备的类型才能针对对它输出对应的内容。现在市面上已经有非常成熟的兼容大量设备的 User Agent 特征库和探测工具，[这里有 Mozilla 整理的一个列表](#)。其中，既有运行在浏览器端的，也有运行在服务端代码层的，甚至有些工具提供了 Nginx/Apache 的模块，负责解析每个请求的 UA 信息。

实际上我们推荐最后一种方式。基于前后端分离的方案决定了 UA 探测只能运行在服务器端，但如果把探测的代码和特征库耦合在业务代码里并不是一个足够友好的方案。我们把这个行为再往前挪，挂在 Nginx/Apache 上，它们负责解析每个请求的 UA 信息，再通过如 HTTP Header 的方式传递给业务代码。

这样做有几点好处：

1. 我们的代码里面无需再去关注 UA 如何解析，直接上层取出解析后的信息即可。
2. 如果在同一台服务器上有多应用，则能够共同使用同一个 Nginx 解析后的 UA 信息，节省了不同应用间的解析损耗。



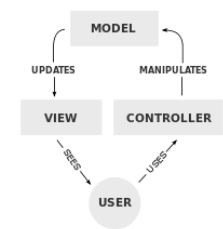


淘宝的 Tengine Web 服务器也提供了类似的模块 [ngx\\_http\\_user\\_agent\\_module](#)。

值得一提的是，选用 UA 探测工具时必须要考虑特征库的可维护性，因为市面上新增的设备类型越来越多，每个设备都会有独立的 User Agent，所以该特征库必须提供良好的更新和维护策略，以适应不断变化的设备。

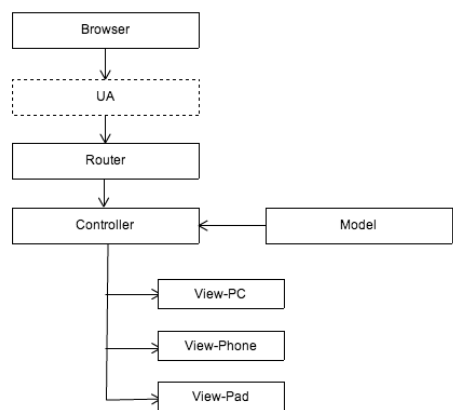
### 建立在 MVC 模式中的几种适配方案

取得 UA 信息后，我们就要考虑如果根据指定的 UA 进行终端适配了。即使在 NodeJS 层，虽然没有了大部分的业务逻辑，但我们依然把内部区分为 Model / Controller / View 三个模型。



我们先利用上面的图，去解析一些已有的多终端适配方案。

#### 建立在 Controller 上的适配方案

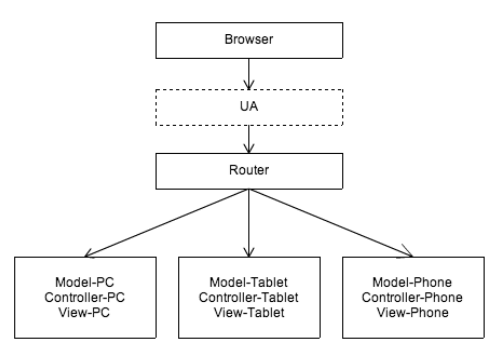


这种方案应该是最简单粗暴的处理方法。通过路由 (Router) 将相同的 URL 统一传递到同一个控制层 (Controller)。控制层再通过 UA 信息将数据和模型 (Model) 逻辑派发到对应的展现 (View) 进行渲染，渲染层则按预先的约定提供了适配几个终端的模板。

这种方案的好处是，保持了数据和控制层的统一性，业务逻辑只需处理一次遍可以应用在所有终端上。但这种场景只适合如展示型页面等低交互型的应用，一旦业务比较复杂，各个终端的 Controller 可能有各自的处理逻辑，如果还是共用一个 Controller，会导致 Controller 非常的臃肿而且难以维护，这无疑是一个错误的选择。

#### 建立在 Router 上的适配方案

为了解决上面遇到的问题，我们可以在 Router 上就将设备区分，针对不同的终端分发到不同的 Controller 上：

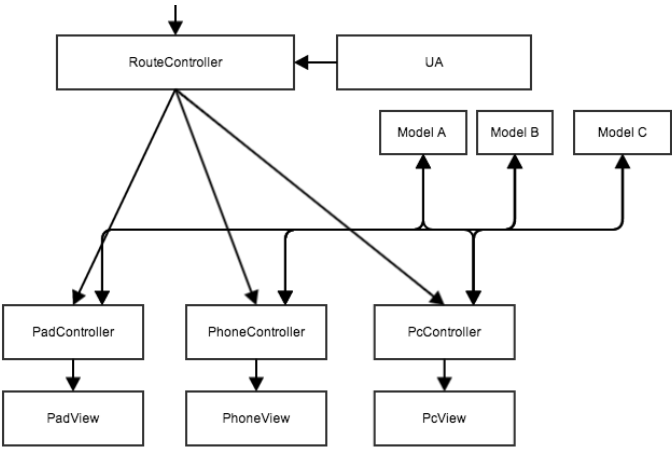


这也是最常见的方案之一，大多表现在针对不同终端使用各自独立的一套应用。如 PC 淘宝首页和 WAP 版的淘宝首页，不同设备访问 [www.taobao.com](#)，服务器会通过 Router 的控制，重定向到 WAP 版的淘宝首页或者 PC 版的淘宝首页，它们各自是完全独立的两套应用。

但这种方案无疑带来了数据和部分逻辑无法共用的问题，各种终端之间无法分享同一份数据和业务逻辑，产生大量重复性工作，效率低下。

为了缓解这个问题，有人提出了优化后的方案：依然是在同一套应用里面，各个数据来源抽象成各个 Model，提供给不同终端的 Controller 组合使用：



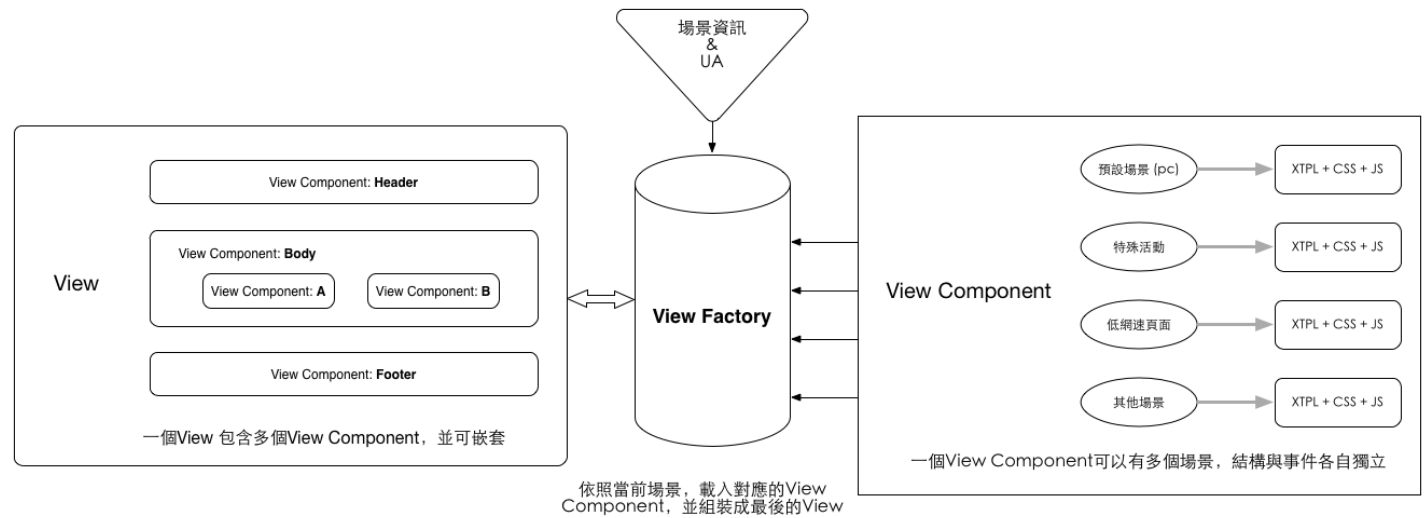


这个方案解决了前面数据无法共用的问题。在 Controller 上各个终端还是相互独立，但能共同使用同一批数据源，至少在数据上无需再针对终端类型开发独立的接口了。

以上两种基于 Router 的方案，由于 Controller 的独立，各个终端可以为自己的页面实现不同的交互逻辑，保证了各终端自身足够的灵活性，这也是为什么大部分应用采用这种方案的主要原因。

**建立在 View 层的适配方案**

这是淘宝下单页面使用的方案，不过区别是下单页将整体的渲染层放在了浏览器端，而不是 NodeJS 层。不过无论是浏览器还是 NodeJS，整体设计思路还是一致的：



在这个方案里面，Router、Controller 和 Model 都无需关注设备信息，终端类型的判断完全交给展现层来处理。图中主要的模块是「View Factory」，Model 和 Controller 将数据和渲染逻辑传递过来之后，通过 View Factory 根据设备信息和其它状态（不仅仅是 UA 信息、也可以是网络环境、用户地区等等）从一堆预设好的组件（View Component）中抓取特定的组件，再组合成最终的页面。

这种方案有几个优势：

- 1. 上层无需关注设备信息（UA），多终端的视频还是交由和最终展现最大关系的 View 层来处理；
- 2. 不仅仅是多终端适配，除了 UA 信息，各个 View Component 还可以根据用户状态决定自身输出何种模版，如低网速下默认隐藏图片、指定地区输出活动 Banner。
- 3. 每个 View Component 的不同模版间可以自行决定是否使用同一份数据、业务逻辑，提供十分灵活的实现方式。

但明显的是，这个方案也是最复杂的，尤其是要考虑一些富交互的应用场景时，Router 和 Controller 也许无法保持这么纯粹。特别对于一些整体性比较强的业务，本身无法被拆分成组件，这种方案也许并不适用；而且对于一些简单的业务，使用这种架构可能不是最佳的选择。

总结

以上几种方案，都各自体现在 MVC 模型中的一个或多个部分，在业务上如果一个方案不满足需求，更可以采取多个方案同时采用的方式。或是可以理解为，业务上的复杂度和交互属性决定了该产品更适合采用哪种多终端适配方案。

对比基于浏览器的响应式设计方案，因为绝大部分终端探测和渲染逻辑迁移到了服务端，所以在 NodeJS 层进行适配无疑带来了更好的性能和用户体验；另外，相对于一些所谓的「云适配」方案带来的转换质量问题，在基于前后端分离的「定制式」方案中也不会存在。前后端分离的适配方案在这些方面有着天然优势。

最后，为了适应更灵活的强大的适配需求，基于前后端分离的适配方案将会面临更多挑战！

- 【附】相关文章列表
- 1. [《前后端分离的思考与实践（一）》](#)
  - 2. [《前后端分离的思考与实践（二）》](#)
  - 3. [《前后端分离的思考与实践（三）》](#)
  - 4. [《前后端分离的思考与实践（四）》](#)

👍 赞

🔖 3 收藏

💬 评论





相关文章

- [dotnet core开发体验之开始MVC](#)
- [Web 开发在 2015 年及未来的发展趋势](#)
- [前后端分离的思考与实践（二）](#)
- [跟开源学Spring MVC：第一章 Web MVC简介](#)
- [四种 JavaScript 客户端 MVC 框架综述](#)
- [利用 Bootstrap 进行快速 Web 开发](#)

可能感兴趣的话题

- [毕业就失业，Finally It comes to me](#) · [👁 21](#)
- [对于互联网公司拖欠延期发工资怎么看待？](#) · [👁 15](#)
- [压力好大，有时候真的想回到老家找个普普通通的工作，不再城市里漂泊了](#) · [👁 22](#)
- [后端转前端，转型期真心痛苦](#) · [👁 14](#)
- [工作满一年，工资一毛没涨是什么样的体验？](#) · [👁 82](#)
- [女生搞开发能干一辈子吗？有没有发展前景啊？](#) · [👁 45](#)

登录后评论

新用户注册

直接登录



文章

输入搜索关键字

搜索



- [本周热门文章](#)
- [本月热门文章](#)
- [热门标签](#)

0 [无休止加班的成因](#)

1 [我见过最有趣的代码注释，都在这里了...](#)

2 [PM 叫你去改一个 Bug，后来.....](#)

3 [HTTPS 工作原理和 TCP 握...](#)

4 [LOL设计模式之「策略模式」](#)

5 [.Net 高效开发之不可错过的实用工具](#)

6 [C++ 及标准库中的那些大坑](#)

7 [算法系列（2）：三只水桶等分水问题](#)

8 [C++ 11 新特性之随机数库](#)

9 [算法系列（1）：Google方程式](#)



业界热点资讯

[更多 »](#)



[用鲁文准备将NetBeans交给Apache管理](#)  
21 小时前 · [👁 3](#)



[让计算机崩溃的简短代码](#)  
3 天前 · [👁 21](#)



[PHP 7.1 新特性一览](#)  
1 天前 · [👁 3](#)



[redis3.2新功能 - GEO地理位置命令介绍](#)  
2 天前 · [👁 4](#)



[JS 又是第一编程语言：GitHub 2016 年度报告](#)  
3 天前 · [👁 34](#) · [👁 3](#)

[精选工具资源](#)

[更多资源 »](#)



[android-remote-notifications: 从远程JSON文件拉取...](#)  
[Android, 通知](#)



[PHP CPP: 一个开发PHP扩展的C++库](#)  
[PHP, 扩展](#)



[Apache Mahout: 经典机器学习算法库](#)  
[Java, 机器学习](#)











[WilliamChart: 优美和直观的图表库](#)  
[Android, 图表](#)



[Singularity: 方便部署和操作的Mesos框架](#)  
[Java, 集群管理](#)

最新评论

-  Re: [我见过最有趣的代码注释，都在这...](#)  
12不是很信
-  Re: [PM 叫你去改一个 Bug，后来...](#)  
现在初三，因为这个原因，理想就是不受雇于人，有一家自己的（小）公司。这种活法太恶心
-  Re: [你应该知道的计算机网络知识](#)  
同意，因为一般一台机器只配有一个网络适配器(网卡)，所以也就默认一个MAC地址对应一台唯一的电脑...
-  Re: [133 行代码实现质感地形](#)  
应该感谢博主搜集了这么好的资源！！辛苦啦 -
-  Re: [133 行代码实现质感地形](#)  
谢谢提醒。已经在正文更新了 :)
-  Re: [PM 叫你去改一个 Bug，后来...](#)  
很赞的文章，翻译得也非常赞。解决客户的问题，这个是最大的价值，当然也不必要放弃原则：P
-  Re: [133 行代码实现质感地形](#)  
你的连接错啦！！现在修改了地址的 <http://demos.playfuljs.com/terrai...>
-  Re: [加入伯乐在线专栏作者，扩大知名...](#)  
MD 的教程，网络上有的，你自己搜搜。

关于伯乐在线博客

在这个信息爆炸的时代，人们已然被大量、快速并且简短的信息所包围。然而，我们相信：过多“快餐”式的阅读只会令人“虚胖”，缺乏实质的内涵。伯乐在线内容团队正试图以我们微薄的力量，把优秀的原创文章和译文分享给读者，为“快餐”添加一些“营养”元素。

- 快速链接
- [网站使用指南 »](#)
  - [问题反馈与求助 »](#)
  - [加入我们 »](#)
  - [网站积分规则 »](#)
  - [网站声望规则 »](#)

关注我们

新浪微博: @伯乐在线官方微博  
RSS: [订阅地址](#)  
推荐微信号



[程序员的那些事](#)   [UI设计达人](#)   [极客范](#)

合作联系  
Email: [bd@jobbole.com](mailto:bd@jobbole.com)  
QQ: 2302462408 (加好友请注明来意)

更多频道

- 小组 - 好的话题、有启发的回复、值得信赖的圈子
- 头条 - 分享和发现有价值的内容与观点
- 相亲 - 为IT单身男女服务的征婚传播平台
- 资源 - 优秀的工具资源导航
- 翻译 - 翻译传播优秀的外文文章
- 文章 - 国内外的精选文章
- 设计 - UI网页，交互和用户体验
- iOS - 专注iOS技术分享
- 安卓 - 专注Android技术分享
- 前端 - JavaScript, HTML5, CSS
- Java - 专注Java技术分享
- Python - 专注Python技术分享



