**关键词：angular,8分钟，组件，代码构筑学**

**使用AngularJS进行组件开发6个月后的五大好处**

**作者：Jurgen Van de Moere 发表日期：2014年11月22日**

2014年2月，谷歌公司宣布，他们已经成功开发并推出一些崭新的“Angular App结构的最佳实践规范”。

这几年来，我一直使用controllers,services,directives,filters等程序目录来构建AngularJS的代码，以及使用初始的AngularJS 文件进行轮廓概述。而现在，这个新的文档突然建议用一个截然不同的方法去构建我们的AngularJS应用程序。

我作为首席开发人员的日常工作中，在尽可能为客户创造最大金钱价值和利益的目标下，（远程）确保所有团队成员力所能及地发挥自身的价值，从而在最短的总时间内使项目达到最佳效果是我的职责。

所以，为了获得新项目的启动并使其快速高效地运转，我先前已经投入了大量的时间去创立一个Yeoman生成器的工具箱和Grunt/Gulp构造过程及文件。虽然维护起来有点痛苦和费力，但是最终起到了作用，尤其是，相比于之前，它运转较快。

另一方面，这个新的规范起到了很大的作用，而且实际上，它帮忙解决了一些我在之前AngularJS应用程序的开发、部署和维护中所重复遇到的问题。其他队员也经常在最后阶段请教我如何部署这些应用程序的代码，因为他们不是十分确定这些应该放在哪里、应该如何部署。

举个例子，通常，我们最后完成的工程文件会以如下熟悉的程序目录进行排列：

my-traditional-app/

|—— public

| |——css

| |——imp

| |——js

| |\_\_\_\_partials

|——src

| |——js

| | |——controllers

| | |——directives

| | |——filters

| | |\_\_\_\_services

| |\_\_\_\_\_\_less

|\_\_\_\_\_\_test

|——e2e

|\_\_\_\_\_unit

|——controllers

|——directives

|——filters

|\_\_\_\_services

在上面程序目录中，当一个工程运行时，每个程序目录包含了几十个甚至几百个文件。

可以想到，我们真的花了很大力气去建立这个看上去合乎情理的分层程序目录结构和尽可能地使得文件的名字具有描述性和表达性。但是，实际上，这样会使程序目录变得难以扩展内容及维护，特别是当你必须在你几个月前已经开发完成的工程中更新代码时，会清晰地感到辛苦费力。

如果连你自己都不能在自己开发的工程中定位一片所找的代码，怎么能期望别人能找到这片代码呢？

所以，在6个月前，我决定全身心作出投入、摒弃我目前正在使用的工具箱，每天早上4点准时起床，在上班前试验各种原型，并开始在新的工程中应用新的规范。

这需要冒非常大的风险，因为整个团队的工作都在依靠我的直觉在引导。但是，今天我可以肯定的告诉你，这个大胆尝试的结果简直让人惊喜万分！

本文接下来的内容，我将和你分享在我们把新规范应用到实践的过程中所遇到的一些挑战，以及我们是如何克服这些挑战的。

 如果你对使用AngularJS进行组件开发一无所知，那么请紧跟我的脚步，然后我会尽我最大的能力向你解释它到底是什么以及介绍我们如何在实践中应用它。

**所以，这新的谷歌规范是关于什么的呢？**

**\*它与真实意义上的代码无关**

这个新的规范并不是关于你应该怎样书写你真实意义上的代码的。



为了详细说明JavaScript/AngularJS的代码编写准则，程序员群体作出了巨大而惊人的努力，其中包括：

\*“Google JavaScript Style Guide”

\*“AngularJS Styleguide”，作者：@ToddMotto

\*“AngularJS Style Guide”，作者：@John Papa

**\*它是关于文件的系统结构的规范**

新的规范建议用如下这种方式构建你的应用程序：把代码进行分组，然后相关的代码群集中进相应的逻辑单元，我们把这些逻辑单元就叫做组件。

这样一来，我们会把文件放入与它们结构上相关的组件目录，而不是把他们放入以其作用（views,controllers,directives等）为特征而编写建立的目录。

结果，一个结构如：

|——public

| |——. . .

| |\_\_\_\_partials

| |——login

| |——form.html

| |\_\_\_\_login.html

|——src

| |\_\_\_\_\_js

| |——controlers

| | |——LoginCtrl.js

| | |——LoginFomCtrl.js

| |\_\_\_\_services

| |——auth.js

| |——session.js

|\_\_\_\_\_test

|\_\_\_\_unit

|\_\_\_\_\_services

|\_\_\_\_\_login

|——auth.spec.js

|\_\_\_\_session.spec.js

变成了另一个结构，如下所示：

|\_\_\_\_\_components

|\_\_\_\_\_login

|——auth.services.js

|——auth.service.spec.js

|——session.service.js

|——session.service.spec.js

|——form.html

|——login.html

|——LoginCtrl.controller.html

|——LoginFormCtrl.controller.html

从第二个结构中我们可以看到，所有与登陆相关的文件都被集中在components/login目录中而不是分散在程序目录的各个地方。



同上，我们可以在在更多的例子中检查原始的“Angular App结构的最佳实践规范”文档以及发现隐藏在这个结构后的详细技术动机和目的。

**对这个规范的第一印象**

首先，你可能没能够立刻看到这个新规范带来的效果和价值。我知道自己开始时并没有看到它的价值。

甚至，完全相反地：它几乎使我觉得登陆组件目录现在犹如已经变成了一个在作用上完全无相关性的文件的聚集地。

但是，我倔强地继续使用着新的规范并且重建了我目前正在开发的一个网页的应用程序，然后，conponents目录最后看起来是这样的：

|\_\_\_\_\_components

|——data

|——header

|——homepage

|——footer

|——login

|——news

|\_\_\_\_portfolio

我不能确定我是否做得正确，但这确实是我应用新规范的第一个工作原型。

一个直接的好处是，当我们观察components目录时，一个有用的印象会浮现在我们的脑海中：在这个应用程序中，哪些东西真正被执行。

但是首先，它证明了找到一片所需的代码是简单方便的。

当我问组里一位设计者如果他必须要更新顶部导航样式的时候他会期望更新哪些点，他看了下组件列表表示可能会先更新header组件。

他说得非常对！

那一个微小的瞬间是对我的一个重要证明与肯定，证明我前进在正确的方向上。

尽管距离做出一些有价值的、能投入生产领域的产品我还有一段很长的路要走，但是这是一个充满希望的开始，使我有足够的动力坚持每天早上4点起床。

**快速前进，回到目前**

从那时以来，我已经正确、成功地试验了多种多样的途径和技术，从而使得能以这一种结构完成工程，这种结构同时也证明了其能在生产环境和领域中取得非常大的成功。

**目前为止的五大好处**

除了在“Angular App结构的最佳实践规范”中概述的技术好处外，还有以下五大到目前为止我们在开发过程中所体验到的好处：

**\*戏剧性般地改善提高了应用程序的设计与构筑**

现在，尽管我们其中一个应用程序包含了超过五十个独立的组件，但即使对于刚加入工程的新开发者来说，弄清楚这个应用程序的作用以及定位代码的位置一件仍然是轻而易举的事情。

**\*提升了代码的可重用性**

我们现在创造大多数组件的方式，是尝试使其能够应用各种不同的应用程序中，同时不需要改动其任何一行的代码。甚至，我们能够在一个Microsoft Windows 8.1 WinRT的应用程序中使用这些组件。

**\*能够即时插入和即时使用组件**

为了把一个我们已经创立的组件添加进一个新的应用程序中，我们随后把这个组件复制到新程序的components目录，然后我们成功了！非常神奇地，它开始了运作！所有的脚本、样式表以及静态资源自动变得有效、可利用。

**\*能轻易地移除组件**

移除组件变成一件轻而易举的事。只需删除目录，然后所有的脚本、样式表规则以及静态资源就会自动被移除，不会留下任何的痕迹，从而（剩下的）代码能一直被保持得简洁干净。

**\*允许更好的团队合作**

事实上，因为每个组件在它自己本身的目录中均是独立的，所以每个团队成员能够在同一个工程中更加有效率地共同工作。并且，极大地减少程序合并时的冲突和使得整个工程的开发过程追踪变得轻而易举。

**结 论**

使用AngularJS进行组件开发仍然处于早期探索的阶段，但它现在正以难以置信的速度发展着。

它不仅给我们带来了许多技术上的益处，而且大大地提高了生产效率，同时提供了大量能使各个规模的团队受益的实用性优点。

正如任何一个构筑设计概念一样，熟练舒适地运用它需要一点时间。但是，一旦你去实践和尝试了，获得的收益将是令人惊喜的。

如果你还没有尝试过，我会强烈推荐你体验一下谷歌公司的“Angular App结构的最佳实践规范”。

**接下来将关注什么呢？**

由于使用AngularJS进行组件开发是一个关于代码构筑学的话题，所以我特意没有在这篇文章中涉及和谈论真实的代码。

然而，在下一篇文章中，我将深入钻研实际意义上的代码，并且谈论我们目前在个人目录中使用的结构以及我第一次尝试失败的原因和过程。

敬请继续关注，同时祝你阅读愉快！

当下一篇文章发表的时候，如果你希望我及时通知你，就请随意在下面留下你的邮箱地址。