ExtJS和AngularJS比较

原文地址：http://www.techferry.com/articles/ExtJS-vs-AngularJS.html

ExtJS 和AngularJS 是两种企业级的富 UI设计框架。 这篇文章从优势、架构、测试、性能等各个方面比较了 ExtJS 和 AngularJS

我们用ExtJS和AngularJS开发了一个简单测试的应用程序，进行完整的性能测试。我们选取了同样的组件，规范了测试的流程，以保证测试的结果。

ExtJS 对比 AngularJS:

- 特性比较
- 性能比较
- 框架比较
- 架构比较
- 进一步的分析

Sencha Ext JS

- 强大的企业级开发框架
- 良好的浏览器兼容性
- 基于MVC的框架开发
- 图标插件
- Modern UI widgets

AngularJS

- 对于 web apps的HTML的增强
- 可以与其他的类库协同工作
- 开源的javascript框架，google维护
- 声明式的编程方法

Sencha ExtJS 和 AngularJS对比

序号	Features	ExtJS	AngularJS
1.	UI应用程序商店	✔	✘
2.	丰富的UI组件样式	✔	幸运的是， AngularUI中已经包含了许多开源的组件， AngularUI Bootstrap, Kendo (and Angular-Kendo), Wijmo 和其他的.
3.	开发单页应用	✔	✔
4.	跨浏览器兼容性	✔	AngularJS 基于jqLite/jQuery实现浏览器的兼容。但是其集成的第三方组件不一定支持跨浏览器兼容。 注意： AnugularJS (1.3 及其以上版本)移除了对IE8的支持。如果需要对于IE8的支持，请选择ExtJS
5.	绘图组件	ExtJS 包含独立的绘图组件，非常的强大。在大部分的企业应用中已经足够	AngularJS D3 绘图组件是基于亚马逊提供的D3 绘图组件。详情请见D3 charts at D3.js.
6.	学习成本	高	中等
架构			
7.	应用程序设计框架	ExtJS 5 支持 MVC (Model-View-Controller) 和 MVVM (Model-View-ViewModel)	MVW (Model-View-Whatever). 它支持流行的设计模式，比如 Model-View-Controller(MVC) 和 Model-View-ViewModel(MVVM)
8.	依赖注入	Ext JS 能够通过扩展（ Deft JS）增加依赖注入	✔
9.	数据绑定	✔	✔
10.	直接操作DOM元素	✔	✔
11.	模块化	✔	✔
测试			
12.	自动测试支持	通过扩展工具实现	✔
13.	测试框架或者测试工具	许多第三方的测试框架，比如 Siesta (专门为ExtJS优化), Jasmine and Mocha 能够用于ExtJS的测试	AngularJS 自带了Karma用于端对端的测试。 Protractor 是用于 Angular apps端对端测试的框架
性能			
14.	Dom算法	深度优先，自底向上	对于DOM树，指令是深度优先、自底往上的算法。而对于控制器则是自上往下的方式。
15.	性能	高度的封装，所以 ExtJS 相对较慢	在我们的性能测试中， AngularJS 比ExtJS快3倍。
16.	轻量级 - 下载的包	✘	✔
移动端			
17.	Web应用的移动端支持	Sencha 建议 Sencha Touch开发移动端版本。有些组件，比如NestedList，在移动端，比Grids要好。这就意味着，需要开发独立的移动版应用。	使用ng-touch库、angular-gestures 和 angularJS 响应模块实现响应式Web应用，
18.	跨平台的移动应用或者混合应用	使用 Sencha Touch 和 Apache Cordova/Phonegap 集成来实现	AngularJS使用 Trigger.io, Cordova/Phonegap integration, Ionic framework 来开发丰富和健壮的移动应用
19.	移动站点	Sencha touch就是用来开发移动站点的	AngularJS 响应模块(angular-responsive, angular-deckgrid 等)， UI Bootstrap， AngularJS responsive directives, angular-gestures 和 ngTouch库.
路由			
20.	内建路由	✔ ExtJS 5已包含	✔
21.	深度链接	✔ ExtJS 5已包含	✔
22.	浏览历史， 前进和后退的支持	✔ ExtJS 5已经引入 在早期的版本中通过Ext.util.History实现	✔
23.	浏览器收藏夹支持	✔ ExtJS 5已引入	✔
24.	SEO 支持	对于大部分的单页应用程序，访问都是基于权限的，所以没有SEO的必要。 对于公开的页面，可以考虑使用 Ajax based SEO。	Consider Ajax based SEO with either Prerender.io or headless browser support in your web-server.
部署			
25.	编译工具	ExtJS 4.x以上版本使用Sencha 命令工具 Sencha sdk 工具用于升级 ExtJS 3.x	第三方的 Grunt工具
26.	包管理工具	Sencha cmd工具	Yeoman, Grunt 和 Bower
Licensing and Support			

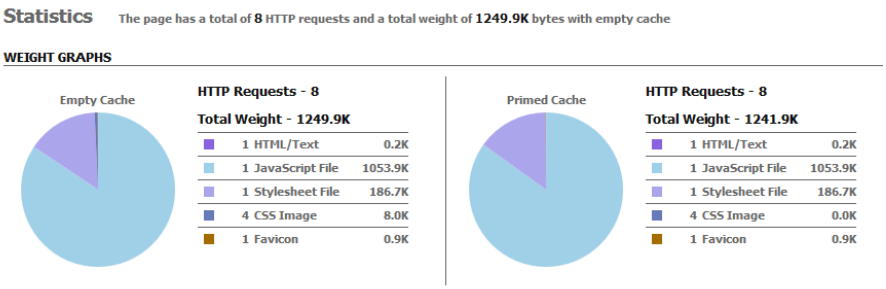
序号	Features	ExtJS	AngularJS
27.	协议	Per-seat / per-server commercial license 和 开源软件基于 GPL 许可	开源的JavaScript 框架，基于 MIT 许可.
28.	完整的稳定，教程，食品，案例	✔	✔
29.	支持：一般的讨论，bug提交，开发新特性请求	基础支持和付费支持	完全支持
30.	完整程度：集成第三方控件，不存在因第三方控件BUG而引发的问题	✔	✘
其他			
31.	动画效果支持	✔	✔
32.	Deferred and Promises	DeftJS provides a number of extensions for ExtJS, including Deferred and Promises.	✔
33.	Dirty Checking	✔	✔
34.	Deferred Bootstrap	Until now, ExtJS applications' testing has never required dela y in the Bootstrap as third-party JavaScript testing framework s which are used for ExtJS application testing do not require d eferring the bootstrap.	Angular Scenario Runner and Batrang requ ire Deferred Bootstrap and hence Deferred Bootstrap concept is introduced to allow en d to end tests.

性能比较

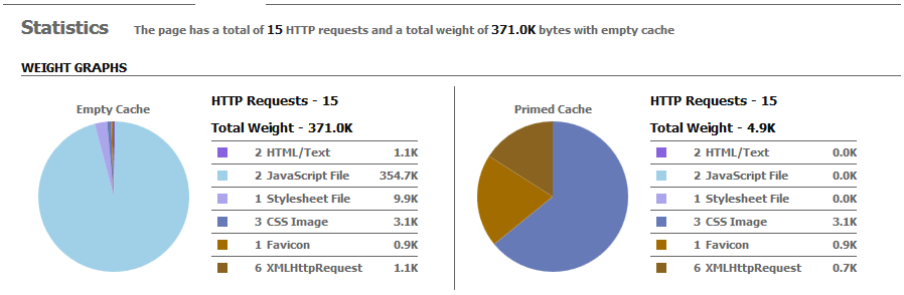
为了全方位比价两个框架的性能，我们开发了一个包含简单的表格，树，绘图组件的应用。规范使用这两种技术构建应用的过程 - 这样我们能够构建一个尽量小的应用来进行深度的测试。 YSlow 用于收集 performance数据。其他方面：

- ExtJS：使用表格、树和绘图组件
- AngularJS: 使用angular-charts.min.js, angular.min.js, jquery.min.js, angular-route.min.js, ng-grid-2.0.7.min.js, angular.treeview.min.js, angular-animate.js modules.
- 相同的http请求，所有数据来源于同一服务器
- 结论： ExtJS 比 AngularJS慢3倍

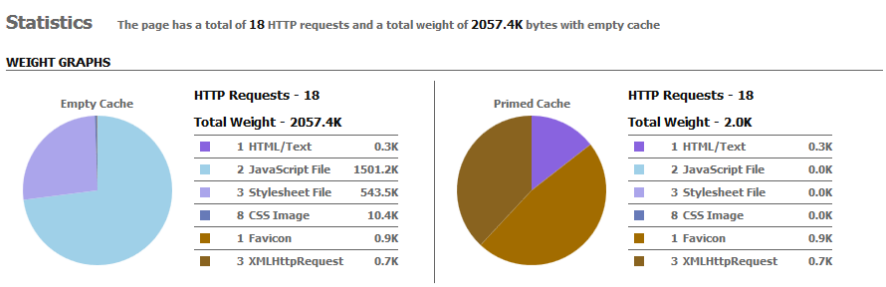
ExtJS:



AngularJS:



If we use ext-all, then the stats are:



两种框架的具体使用环境

使用 ExtJS：

- 你想使用强大的ExtJS组件。这会节约很多的时间。
- 你不想担心兼容性问题。这是一大亮点
- 你和你的客户需要一个专门的许可或者需要专门的支持
- 如果桌面应用和移动应用是两套程序。
- 如果愿意支付支持费用

使用 AngularJS：

- 要求使用更小的包
- 要求响应式设计
- 需要自动测试
- 集成第三方组件
- 你的团队能够调试CSS并解决浏览器兼容性问题
- 你的团队能够解决第三方组件的BUG

架构的不同：

我们从11个方面来分析这两种技术的不同.

1. 应用开发框架
2. 组件
3. 路由
4. 测试
5. 数据绑定
6. SEO
7. 移动解决方案
8. Dom 算法
9. Deferred and Promises
10. 脏检查
11. Deferred bootstrap

摘要：

ExtJS 是基于组件的 (表格、树、表单、绘图)；代码开始于对现有类的扩展，配置模型，自定义显示和行为/事件，将这些组件加入到container/layout中。 它遵循面向对象的设计思想和MVC设计模式，很少直接操作DOM元素

AngularJS，走的是另外的一条路，是一种声明式的程序。在HTML标签上新增AngularJS指令，配置模型，使用模版和路由配置视图；框架最终实现了DOM的创建。我们仍然是基于HTML架构，操作DOM元素

应用设计框架

Ext JS：

- 支持 MVC 和 MVVM
- 随着ExtJS 应用规模和复杂度的增加， Sencha Touch 和 Ext JS 能够通过 Deft JS 实现依赖注入
- 它是基于组件，模块化的

AngularJS：

- AngularJS是一个对于web应用的HTML扩展。
- 它被描述为了 'Model-View-Whatever' 框架，它并没有规定使用特定的开发框架。但是它非常容易支持 Model-View-Controller 或者 Model-View-View Model.
- 它是模块化的。 它自动装盘HTML元素中的依赖注入模块

组件：

Ext JS：

ExtJS和其他javascript框架最大的区别，就是拥有一套强大的UI组件
好处：
完整的组件体系，能够大大的减少开发时间
坏处：
ExtJS的组件会生成冗长的DOM代码，这样会影响效率

AngularJS：

- AngularJS 并不具有一个完整的组件库
- 但是，这里有非常多的第三方组件，比如 AngularUI Bootstrap, Kendo (and Angular-Kendo), Wijmo 等等

路由：

ExtJS：

ExtJS 5 支持路由

- ExtJS 5 路由通过使用浏览器历史堆栈，实现了应用程序状态跟踪
- 它允许深度链接到应用程序。 允许直接链接到应用程序的一个部分
- 它支持浏览器收藏夹和浏览器前进/后退导航

AngularJS：

- AngularJS路由将 controllers, view templates, 和浏览器当前 URL 地址联系在了一起。使用这个特性，我们实现了深度链接
- 通过使用深度链接，实现了使用一个超链接，导航到本页面的特定位置。它使我们能够使用浏览器历史（前进/后退导航）和浏览器收藏夹

测试：

ExtJS：

- ExtJS应用能够使用第三方的测试框架来测试，比如 Siesta (专门为 Ext JS 优化)，Jasmine 和Mocha.
- 自身并不包含测试框架和测试工具

AngularJS：

- AngularJS在设计之初，就考虑到了应用的可测试性
- 支持单元测试，集成测试和功能
- AngularJS 团队开发了 Karma 测试工具。 另外，第三方的工具 Protractor 也可以用于AngularJS的测试

数据绑定

双向数据绑定是连接页面UI和数据模型的纽带，及更改模型的数据，页面UI能够同步响应

ExtJS：

- ExtJS 5 组件有新的绑定配置，用于实现此功能
- 在早期的ExtJS版本中，使用Store对象来实现此功能。但是任然有一些工作需要做，比如加载Store等

AngularJS：

AngularJS的双向数据映射执行的作用域，是基于一个原型继承树作用域的嵌套模型 (models).

Cons：
如果模版中有2000-3000个绑定应用程序会变慢

当许多数据呈现在了页面上的时候，Bindonce是一种很好的方法来减少监视。一旦渲染完成，就不会改变，你不需要在对其改变进行监视

SEO：

Most of the single page apps which work behind authentication need not be indexed for SEO. If you have some pages in your app which are public and which needs to be indexed, you can create them separately, either with static HTML/CSS or if you do need to use dynamic content, consider Ajax based SEO as described below.

Ajax based SEO: For the indexing of dynamic / ajax-based single page web apps, all you have to do is to generate the additional static content so that when the crawlers access your page, they get easy access to that static content and when the users access your page, they see the app. To achieve this functionality you could either use some tools like Prerender.io: fully open-source or you have to set up the headless browser support in your web-server which is an additional effort.

ExtJS: Ajax based seo is possible in ExtJS with hashbang urls' support in your web-server.	AngularJS: • AngularJS seo with Prerender.io: When a crawler visits your page at hashbang url, the Prerender service will check and see if it has a snapshot or already rendered page for that URL, if yes, it will send it to the crawler, if not, it will render a snapshot on the fly and send the rendered HTML to the crawler for correct indexing. • Alternatively, you can also build support for hashbang URLs which may require you to set-up your web-server to summon-up the headless html browser.

移动解决方案：

ExtJS：
 	
• Sencha Touch - 企业级高性能的移动HTML5框架，用于开发强大的、健壮的移动APP或者移动站点。
• Sencha touch 集成Cordova/Phonegap，实现了跨平台的混合应用。
	AngularJS:
	
	• AngularJS 能够用于开发响应式的web应用/站点。
	• 为了开发跨平台的混合应用，AngularJS集成了以下组件
	  ◦ Trigger.io
	  ◦ Ionic Framework - Advanced Html5 hybrid mobile framework and optimized for AngularJS
	  ◦ Cordova/Phonegap

Dom算法

ExtJS： 深度优先、自底向上	AngularJS: Directives are linked in a Depth-First, Bottom-Up approach to the DOM tree. Controllers are linked in a top-down manner.

Deferred and Promises:

Deferred and Promises break the complexities of asynchronous programming, separate out the synchronous and asynchronous world, remove the tight coupling between the two They are for asynchronous programming what try, catch and throw keywords are for synchronous programming.

ExtJS: ExtJS augmented with DeftJS may provide Deferred and Promises.	AngularJS: AngularJS offers an implementation of the Q API for Deferred and Promises.

脏检查

脏检查：框架比对旧值和新值，如果他们不一致，就触发change事件

ExtJS： • ExtJS 4.x使用store的binding特性来实现脏检查。 • ExtJS 的store允许你通过设置autoSync为false，来延迟脏检查行为。用户对数据的改变，会在UI上给上举打上一个标记，当更新store的时候，能够批量提交这些更新。 • 将数据的变化放映到UI上，需要做一点点工作，比如reloading store的数据	AngularJS： • Angular 使用Digest Cycle 实现了脏检查 • 使用Angular api，你不需要手动处理，angular 会自动进行脏数据的回收。但是对于第三方的api，你需要调用$apply方法来回收脏数据。 • 脏数据回收完成，数据的改变就会呈现在UI上。 • 脏检查是异步的

Deferred bootstrap:

Bootstrap指的是初始化的过程，Deferred bootstrap用于延迟启动进程加载很重、很大的依赖文件或者目标。 Deferred bootstrap主要用于端对端测试。

虽然deferred bootstrap对于开发和单页应用测试没有意义，但是它对AngularJS 应用的端对端测试很有用。一些javascript测试工具，比如 Batrang 和 Angular Scenario Runner (AngularJS团队开发的，用于AngularJS项目的端对端测试) 需要 deferred bootstrap.

ExtJS: 多种受欢迎的JavaScript 测试框架，比如Siesta (专为 Ext JS 优化), Jasmine 和 Mocha，并不需要延迟EXT JS应用程序的启动	AngularJS: • Batrang是一个新的angular 团队推荐的 Chrome插件，提供了一个显示应用性能瓶颈，应用DEBUG的工具。 • AngularJS Batarang and Angular Scenario Runner require Deferred Bootstrap feature to hook into angular's bootstrap process and sneak in more modules into the DI registry which can replace or augment DI services for the purpose of instrumentation or mocking out heavy dependencies.

AngularJS的消化周期:

- 消化周期指的是数据改变的相应周期
- 一般情况下， 浏览器的event-loop等待事件到达，当它接收到事件，会在输入控件上发出事件，对于的事件句柄会捕获到此事件，触发对应的函数，将其作为参数传入AngularJS的执行上下文中。
- 函数中将实现模型的改变，在消化周期结束的时候，异常句柄会被触发
- 在所有的消化周期机制中，在观察列表中的观测者会进行迭代，通过每个观测者的表达式，获取到特定的作用域， 作用域中的新值和旧值会进行比较，判断是否改变，然后 对应的函数就会被执行。这时，可能会发生下面两种情况：
    - 如果监听函数没有改变作用域，浏览器会重新标记DOM元素为脏数据，然后模型会被声明为稳定的，消化周期结束。
    - 如果作用域改变中，那么就会触发其他的监听，这时，观测者保持运行状态直到没有其他的观察者被触发。对多能触发10个观测者，否则会抛出'Maximum iteration limit exceeded' 错误
- 脏数据检查请看 asynchronously.

三种脏数据检查机制

- **基于引用的脏数据检查：** 使用===来进行新旧值的比较。这种方式对于内存和计算都效率都是最好的，因为这种方式不需要复制、遍历数据。它只是进行引用比较。
- **基于值的脏数据检查：**它会进行深度-对象-树比较。这意味着，在每一个 $digest 周期， AngularJS 将检查新老数据是否具有相同的结构。
- **基于集合的脏数据检查：** 它比较物理对象引用。集合观察者维护了一个对于数组和对象的值复制,在每个消化周期，遍历新旧值，通过===来比较是否改变。比如，不同于基于引用的脏数据检查， it goes one-level deep and performs an additional, shallow reference check of the top level items in the collection.

脏检查的关键点：
如果一个模型有2000-3000个观测者，那么系统会变慢
就算你的应用足够快，使用户不会感到相应延迟，你也不能直接在一页上展示2000个信息给用户。因为这是一个很坏的UI设计，用户会感动无所适从，但是，使用排序组件或者具有双向绑定功能的表格，你能够很好的组织这2000条数据，给用户更好的展现。

观测者：

默认情况下，所有绑定到UI的模型数据都被观察。比如，他们都有一个观测者注册到了观察列表上。你也可以通过$watch 函数手动触发观测者。一个观测者有两个函数：一个观察函数和一个观察表达式。观察函数是用于数据改变时被触发的监听函数，观察表达式是确定那些数据需要被观察（作用域）。

Asynchronous nature of dirty cycle:

通过赋值，比如 $scope.username="angular" ，并不会马上触发$watch，而是在 $digest 阶段才会触发。这个延迟的好处，是可以将多个模型的更新操作，合并到一个$watch消息通知中，这样，可以确保这个$watch通知运行时，没有其他的 $watches 在允许。如果一个$watch改变了模型的值，它就会被加入$digest周期