**把MEAN工具栈开发的应用部署到云端，开发移动应用**

1. **本地部署：**部署到云端时使用Node.js做服务器，使用ExpressJS应用框架，使用MongoDB做数据库。
2. **安装Node.js、npm和MongoDB:**安装一个支持Node.js的NetBeans插件，配置插件。
3. **使用MEAN工具栈开发应用：**使用MongoDB的API就能与数据库交互，不用编写SQL查询，使用JSON通过API与MongoDB交互。使用MEAN工具栈开发的应用运行在Node.js平台中。
4. **Node.js应用的public文件夹：**AngularJS代码刚在public文件夹中。
5. **为了使用MEAN工具栈修改服务：**每个服务的URL改成./NodeBlog/。
6. **为了使用MEAN工具栈，修改博客应用的控制器：**controllers.js文件实现了发布新博客文章和评论功能。对BlogViewCtrl控制器添加了subumit方法用于处理在博客文章中发布评论的过程，submit方法调用了BlogPostComments服务中的save方法。BlogPostComments服务连接的REST服务需要基本认证，使用$http服务在发给REST服务的请求中添加了一个用于基本认证的首部。调用$location服务的path方法和$route服务的reload方法刷新博客文章页面，显示新添加的评论。再添加一个新的控制器NewBlogPostCtrl的submit方法同时此方法调用了BlogPost服务中的save方法。
7. **为了使用MEAN工作栈，修改博客应用的模板：**NewBlogPostCtrl控制器中有个JSON数组，保存在languageList属性中，生成HTML<select>元素。修改模板添加一个表单用于提交新评论，这个表单对应于BlogEntryCtrl控制器中的submit方法，用户提交表单时，博客文章ID会发给控制器，再发给添加新评论的REST服务。使用ng-submit指令把表单提交操作交给NewBlogPostCtrl控制器中的submit方法处理，再menu.html文件的导航菜单中添加一个链接指向发布新博客文章视图。再app.js文件中添加用于发布博客文章的新路由并添加到$routeProvider中。
8. **添加Node.js依赖：**打开cmd窗口进入NodeBlog项目再硬盘中的位置；输入npm install回车后安装博客应用的所有依赖。
9. **在本地运行博客应用：访问http：//localhost：8080使用用户名：node密码：password登陆应用。**
10. **编写Karma测试规约：详见《AngularJS学习手册》P162-P168代码部分。**
11. **编写Protractor测试规约：详见《AngularJS学习手册》P169-P170代码部分。**
12. **把使用MEAN工具栈开发的应用部署到云端步骤操作如下：**

****



打开cmd窗口，进入OpenShift所在文件夹，执行npm install命令，在项目名称右键从弹出菜单中选择Run，如果指示器显示正在运行说明应用能正常运行。

**AngularJS基础：**

AngularJS 是一个 JavaScript 框架。它是一个以 JavaScript 编写的库。AngularJS 是以一个 JavaScript 文件形式发布的，可通过 script 标签添加到网页中：<script src="http://cdn.static.runoob.com/libs/angular.js/1.4.6/angular.min.js"></script>

**AngularJS表达式：**可以包含字母，操作符，变量。 AngularJS 表达式可以写在 HTML 中。AngularJS 表达式不支持条件判断，循环及异常。AngularJS 表达式支持过滤器。

**AngularJS 指令：**

AngularJS 指令是扩展的 HTML 属性，带有前缀 **ng-**。

* **ng-app** 指令初始化一个 AngularJS 应用程序。**ng-app** 指令定义了 AngularJS 应用程序的 **根元素**。**ng-app** 指令在网页加载完毕时会**自动引导**（自动初始化）应用程序。
* **ng-init** 指令初始化应用程序数据。**ng-init** 指令为 AngularJS 应用程序定义了 **初始值**。通常情况下，不使用 ng-init。您将使用一个控制器或模块来代替它。
* **ng-model** 指令把元素值（比如输入域的值）绑定到应用程序。**ng-model** 指令也可以：

1. 为应用程序数据提供类型验证（number、email、required）。
2. 为应用程序数据提供状态（invalid、dirty、touched、error）。
3. 为 HTML 元素提供 CSS 类。
4. 绑定 HTML 元素到 HTML 表单。

**AngularJS 应用组成如下：**

* View(视图), 即 HTML。
* Model(模型), 当前视图中可用的数据。
* Controller(控制器), 即 JavaScript 函数，可以添加或修改属性。

**AngularJS 应用组成如下：**

AngularJS 过滤器可用于转换数据：

1. 1、uppercase，lowercase 大小写转换

{{ "lower cap string" | uppercase }} // 结果：LOWER CAP STRING

{{ "TANK is GOOD" | lowercase }} // 结果：tank is good

2、date 格式化

{{1490161945000 | date:"yyyy-MM-dd HH:mm:ss"}} // 2017-03-22 13:52:25

3、number 格式化（保留小数）

{{149016.1945000 | number:2}}

4、currency货币格式化

{{ 250 | currency }} // 结果：$250.00

{{ 250 | currency:"RMB ￥ " }} // 结果：RMB ￥ 250.00

5、filter查找

输入过滤器可以通过一个管道字符（|）和一个过滤器添加到指令中，该过滤器后跟一个冒号和一个模型名称。

filter 过滤器从数组中选择一个子集

// 查找name为iphone的行

{{ [{"age": 20,"id": 10,"name": "iphone"},

{"age": 12,"id": 11,"name": "sunm xing"},

{"age": 44,"id": 12,"name": "test abc"}

] | filter:{'name':'iphone'} }}

6、limitTo 截取

{{"1234567890" | limitTo :6}} // 从前面开始截取6位

{{"1234567890" | limitTo:-4}} // 从后面开始截取4位

7、orderBy 排序

// 根id降序排

{{ [{"age": 20,"id": 10,"name": "iphone"},

{"age": 12,"id": 11,"name": "sunm xing"},

{"age": 44,"id": 12,"name": "test abc"}

] | orderBy:'id':true }}

// 根据id升序排

{{ [{"age": 20,"id": 10,"name": "iphone"},

{"age": 12,"id": 11,"name": "sunm xing"},

{"age": 44,"id": 12,"name": "test abc"}

] | orderBy:'id' }}

**AngularJS 服务：**在 AngularJS 中，服务是一个函数或对象。

**$location** 服务：它可以返回当前页面的 URL 地址。

**$http** 服务：AngularJS 应用中最常用的服务。 服务向服务器发送请求，应用响应服务器传送过来的数据。

**$timeout** 服务：对应了 JS **window.setTimeout** 函数。

**$interval** 服务：对应了 JS **window.setInterval** 函数。