目录

[一.NPM 2](#_Toc504390484)

[1.1 简介 2](#_Toc504390485)

[二.Bower（没必要用） 2](#_Toc504390486)

[三.Gulp 2](#_Toc504390487)

[3.1简介 3](#_Toc504390488)

[3.2准备工作 3](#_Toc504390489)

[3.3基本使用 3](#_Toc504390490)

[3.3.1 常用插件 3](#_Toc504390491)

[四.git 4](#_Toc504390492)

[4.1 介绍 4](#_Toc504390493)

[4.2 安装git 4](#_Toc504390494)

[4.3 git命令操作 4](#_Toc504390495)

[五.angularJs（简称ng） 4](#_Toc504390496)

[六. MVC（以angular为例） 6](#_Toc504390497)

[6.1 模块 6](#_Toc504390498)

[6.1.1$watch 8](#_Toc504390499)

[$scope(上下文模型) 9](#_Toc504390500)

[6.2 指令（Directive） 9](#_Toc504390501)

[6.2.1 ng-app 9](#_Toc504390502)

[6.2.2 ng-bind 9](#_Toc504390503)

[6.2.3 ng-repeat 9](#_Toc504390504)

[6.2.4 ng-cloak 10](#_Toc504390505)

[6.2.5 ng-hide，ng-show，ng-if 10](#_Toc504390506)

[6.2.6 ng-href 、ng-src、ng-switch 12](#_Toc504390507)

[6.2.7 其他指令 13](#_Toc504390508)

[6.2.8 自定义指令（用的非常少） 14](#_Toc504390509)

[7.DOM2 15](#_Toc504390510)

[7.1可以绑定多个元素。传统：只能绑定一个事件 15](#_Toc504390511)

[7.2动态的添加和移除事件 15](#_Toc504390512)

[7.3 事件委托 15](#_Toc504390513)

# 一.NPM

## 1.1 简介

1）什么是NPM

-Node package Manager

-Node应用程序依赖包的管理工具

-安装卸载更新之类的操作

2）为什么使用NPM

-包很多

-场景：我需要用一个A，A依赖B，B依赖C

-常见的包管理工具都有循环依赖的功能

-你只需要记住你要什么东西

# 二.Bower（没必要用）

1）什么是Bower

-官网：<http://bower.io>

-web应用程序依赖项管理工具

2）为什么使用

-方便便捷的方式管理包

3）安装

-npm install -g bower //-g:global

# 三.Gulp

当下的前端开发

-不再是简简单单的使用HTML + CSS + Javascript这些简单的技术构建网页应用程序了

-我们要提高效率，就必须减少重复的工作

-使用Less之类的预处理的CSS coffeesscript

-提供开发阶段的便利，开发阶段更快捷

现在的开发行业优质的开发人员是不应将精力放在这些重复性质的工作上

-Gulp就是一种可以自动化完成我们开发过程中大量的重复工作

+预处理语言的编译

+js css html压缩混淆

+图片体积优化

-除gulp之外还有一些类似的自动化工具，比如grunt

## 3.1简介

-当下最流行的自动化工具

+什么是自动化构建工具？

+自动完成一系列重复的操作

-链接：

+官网：<http://gulpjs.com>

+中文网：<http://www.gulpjs.com：cn>

-gulp即包含工具，还有一些编程的API

-就是用来机械化的完成重复性质的工作

-gulp的机制就是将重复工作抽象成一个个的任务

## 3.2准备工作

-安装Node.js

+http://npm.taobao.org，淘宝镜像

+安装后可以通过命令行工具node -v查看版本

-安装gulp命令行工具

+npm install -g gulp

-初始化gulp项目

-创建任务 – gulpfile.js

## 3.3基本使用

### 3.3.1 常用插件

-编译less：gulp-less：<https://www.npmjs.cpm/package/gulp-less>

-创建本地服务器：gulp-connect：https//www.npmjs.com/package/gulp-connect

-合并文件：gulp-concat：gulp-connect：https//www.npmjs.com/package/gulp-concat

具体用法可以去官网查看

# 四.git

## 4.1 介绍

-什么是git？是一个源代码管理工具

-https://guides.github.com/

## 4.2 安装git

-git命令行工具

-或基于git命令行的一个客户端软件（提供一个界面去管理源码）

## 4.3 git命令操作

-初始化一个本地git仓库

+ git init

+就是在本地文件夹中添加了一个.git文件夹用于记录所有的项目变更信息

-查看本地仓储的变更状态

+git status

+git status -s //输出简要的变更日志

+用于查看本地仓储的状态，第一次查看，显示的是一坨没有被跟踪的文件

-添加本地暂存（托管）文件

+git add（add . 是添加当前所有文件）

+可以讲一个没有被跟踪的文件添加到跟踪列表

-添加忽略跟踪的文件

+在文件夹中添加一个 .gitignore的文件（window不能命名一个点开头的文件，可以用命令行或者sublime创建 ），将文件名卸载里面，就会被忽略

-提交被托管的文件变化到本地仓储

+git commit （-m “第一次提交”） //-m意思是message，最好加上这样的提交日志

+将本地变化提交到本地的仓库文件夹归档

-对比差异

+git diff 用于对比

-提交日志

# 五.angularJs（简称ng）

在开始学习AngularJs之前，只需要具备以下基础知识：HTML、CSS、Javascript，为了更好得学习效果，最好擅长一些JavaScript库，后台用的多一点

-主要思路不是传统的“先查找再操作DOM元素”，而是“以数据为中心”。适用于以数据操作为主的SPA应用

-一款非常优秀的前端高级JS框架

-有了这一类框架就可以轻松构建SPA应用程序（单页应用程序）

+只有一个页面（整个应用的一个载体）

+内容全部是由AJAX方式呈现出来的

-通过指令扩展了HTML，通过表达式绑定数据到HTML

-代领前端进入MV\*时代

-为什么使用AngularJS？

+代领当前世面上框架走向模式化或者架构化

- 相关链接

+http://www.cnblogs.com/powertoolsteam/p/angularjs-introdection.html

+http://www.angularjs.cn/

- 核心特性

+MVC

+模块化

+自动化双向数据绑定

+指令系统

-使用总结

+最大程度的减少了页面上的DOM操作

+让Javascript中专注业务逻辑代码

+通过简单的指令结合页面结构与逻辑数据

+通过自定义指令实现组件化编程

+代码结构更合理

+维护成本更低

+解放了传统JS中频繁的DOM操作

-如何查看本地angular文档？

+在angular怎目录下在命令行输入hs -o即可（要安装http-server，npm命令：npm install  -g http-server）

<**body ng-app ng-init="user.name=**'world'**"**>  
 <**h1**>使用ng实现双边数据绑定</**h1**>  
 <**input type="text" placeholder="输入你的姓名" ng-model="user.name"**>  
 <**p**>hello <**strong**>{{user.**name**}}</**strong**></**p**>  
</**body**>

**ng-init会在开始时绑定一个初始值**

**ng-model绑定一个模型，它是双向绑定数据的指令效果就是将当前元素的value属性和模型中的user-name建立绑定关系**

**-ng-bind:计算表达式**

**用法：可以当成属性来用：<p ng-bind=表达式>**

**也可以当做class名来用：<p class=”ng-bind:表达式”>**

**说明：此指令作用与{{}}表达式基本类似，只是可以防止用户在一瞬间看到{{}}。ngBing指令计算完成表达式的值，会替换当前元素的innerHTML**

**-算术运算 +-\*/%都可以，唯独不能自增++和自减—**

**-不可以使用new关键字创建对象**

**-不可以调用ES全局函数**

**-ngController:调用Controller创建函数，实例化一个控制器对象，指定其作用范围，用法：<ANY ng-controller=“控制器名”>**

**-ngRepeat：用于在View实现循环输出**

**用法：<ANY ng-repeat=“变量名 in 集合对象”>**

**含义：对于集合对象中的每一个元素，依次赋值给指定的变量**

**名，对每次赋值都输出一边当前元素**

**-ngIf:用于在View实现判断输出，为false就不输出了**

**-AngularJS中声明模型数据的方式**

**1）<ANY ng-init=“变量名=值;”>**

**此方法将Model声明在View中，严重违反了MVC模型分工，不推荐使用**

**2）使用Controller创建Model数据-符合MVC模型分工**

**新版本的AngularJS中创建Model的语法：**

**ngAPP=>Module=>Controller=>Model**

**1.声明一个AngularJS的应用程序：ngApp**

**2.创建一个AngularJS自定义的模块：angular.module(“模块名”,[依赖列表])**

**3.在应用中注册自定义模块：ng-app=“模块名”**

**4.在模块中声明Controller函数**

**5.在view中指定Controller对象的作用范围**

**6.在Controller中声明Model数据**

**例：**

<html lang="en" ng-app="myModule">

<body>  
 <div class="container" ng-controller="myCtrl">  
 <h1>angularjs模板</h1>  
 <p>{{ename}}</p>  
 </div>  
  
 <script>  
 angular.*module*("myModule",["ng"]).  
 controller("myCtrl", function($scope){  
 $scope.ename = "老王";  
 });  
 </script>  
</body>

var的变量只是局部变量，只有声明为$scope才可以

上述script中的东西可以分开写

**var** mm = **angular**.*module***("myModule"**,**["ng"])**;  
mm.**controller("myCtrl"**, **function(**$scope**)**{  
 $scope.**ename** = **"老王"**;  
}**)**;

但是不要这么写，因为污染了全局变量

# 六. MVC（以angular为例）

-是一种应用程序的开发思想 ，不是设计模式

-主要目的是为了解决应用程序展示结构、业务逻辑之间的紧耦合关系

-使应用程序的组成分为三个部件，每个部件有自己明确的职责，相互之间没有依赖

-将应用程序的组成分为三个部分：Model（模型：处理数据和业务逻辑），view（视图：以友好的方式向用户战士数据），controller（组织调度相应的处理模型）

## 6.1 模块

angularJS很重要的一个特性就是实现模块化编程，我们可以通过以下方式创建一个模块，对页面进行功能业务上的划分。

-使用自定义的模块才可以依赖别的包里面定义模块，angular定义的默认模块没有依赖任何模块

//创建一个名字叫做MyApp的模块，第二个参数指的是该模块依赖哪些模块  
**var** myApp = **angular**.*module***("MyApp"**,**[])**;

-也可以将重复使用的指令或过滤器之类的做成模块便于复用

-注意必须指定第二个参数，否则变成找到已经定义的模块

例子：

<**div ng-app="MyApp" ng-controller="DemoController"**> <!--与第一个参数名保持一致-->  
 <**h1**>使用ng实现双边数据绑定</**h1**>  
 <**input type="text" placeholder="输入你的姓名" ng-model="user.name"**>  
 <**p**>hello <**strong**>{{user.**name**}}</**strong**></**p**>  
 <**input type="button" ng-click="***show***()"**>  
</**div**>  
<**script**>  
 //注册模块 通过module函数，  
 // 第一个参数是这个模块的名字  
 //！！！第二个参数是这个模块所依赖的模块，  
 //如果不依赖任何模块也必须传递第二参数,  
 //如果没有第二个参数，angular.module就不是创建一个模块  
 //angular.module返回刚刚创建的模块对象  
 **var** app = **angular**.*module***("MyApp"**, **[])**;  
 //app.controller方法用于创建一个控制器,创建的控制器属于myApp模块  
 app.**controller("DemoController"**, **function(**$scope**)**{//参数名就是这个  
 //当控制器执行时会自动执行的函数  
 $scope.**user** = {};  
 $scope.**user**.**name** = **"张三"**;  
 //$scope不仅仅可以往视图中暴露数据，还可以暴露行为  
 $scope.show = **function()**{  
 **console**.log**(**$scope.**user)**;  
 }  
 }**)**;//所有命名要加上Controller

-可以通过angular.module创建一个模块

-angular.module方法传递两个参数才是创建模块，一个参数是获取模块

### 6.1.1$watch

官方的API中提供了一个$scope.$watch方法

**$scope**.$watch**("username"**, **function(**now, old**)**{  
 **if(**now**)**{  
 **if(**now.length < 7**)**{  
 **$scope**.**message** = **"输入格式不合法"**；  
 }  
 **else**{  
 **$scope**.**message** = **""**;  
 }  
 }  
 **else**{  
 **$scope**.**message** = **"请输入用户名"**;  
 }  
}**)**;

## $scope(上下文模型)

-视图和控制器之间的桥梁

-用于在视图和控制器之间传递数据

-利用$scope暴露数据模型

## 6.2 指令（Directive）

-Angularjs有一套完整额、可扩展的、用来帮助Web应用开发的指令集

-在简历DOM期间，和HTML关联着的指令会被检测到，并且执行

-Angularjs中将ng-这种属性称之为指令，其作用就是为DOM元素调用方法、定义行为绑定数据等。

-简单说：当一个Angular应用启动，Angular就会遍历DOM树来解析HTML，根据指令不同，完成不同的操作

### 6.2.1 ng-app

-用来标明一个AngularJS应用程序

-标记在一个AngularJS的作用范围的根对象上

-系统执行时会自动的执行根对象范围内的其他指令

-可以在同一个页面创建多个ng-app节点（不推荐）

-标记的范围尽可能小，性能

## 6.2.2 ng-bind

-将作用域（scope）中的值绑定到元素的innerHTML上，其效果会比通过表达式绑定的方式更友好

-注意，如果绑定的内容有HTML，会自动转义，为了安全（跨站脚本攻击）

-适用于元素少，多的用ng-cloak

### 6.2.3 ng-repeat

ng-repeat指令用来遍历一个数组重复创建当前元素，如：

<**ul class="message"**>  
 <**li ng-repeat="item in message track by $index"**>  
 {{item}}  
 </**li**>  
</**ul**>

### 6.2.4 ng-cloak

-用于在angularjs应用在加载时防止angularjs代码未加载完而出的问题。

-angularjs应用在加载时，文档可能会由于angularjs代码未加载完成而出现显示angularjs代码，进而会有闪烁的效果，ng-cloak就是为了防止该问题发生

-如果angularjs实在head中引入的话没问题

-如果引用在Body的最下面则需要hack

### 6.2.5 ng-hide，ng-show，ng-if

-ng-show = "true"就是永远显示

-ng-show = "false"就是永远不显示

-ng-hide同上

- ng-show决定是否显示，ng-hide是否隐藏，ng-if是否存在DOM元素

例：3s后loading遮盖层消失

.**tips**{  
 **position**: **absolute**;  
 **left**:0;  
 **right**:0;  
 **top**:0;  
 **bottom**:0;  
 **background-color**: **rgba**(100,100,100,.5);  
 **font-size**:40**px**;  
 **line-height**: 100**vh**;  
 **text-align**: **center**;  
 }  
 </**style**>  
</**head**>  
<**body ng-controller="ListController" ng-app="myApp"**>  
 <**div**>  
 aaaaaaaaa  
 </**div**>  
 <**div class="tips" ng-show="loading"**>  
 loading...  
 </**div**>  
 <**script**>  
 **angular**.*module***("myApp"**, **[])** .**controller("ListController"**, **["$scope"**, **"$timeout"**, **function(**$scope, $timeout**)**{  
 $scope.**loading** = **true**;  
 $timeout**(function()**{  
 $scope.**loading** = **false**;  
 }, 3000**)** }**])**;  
 </**script**>

## 6.2.6 ng-href 、ng-src、ng-switch

- ng-href /ng-src用于解决当链接类型的数据绑定时造成的加载BUG

-ng-switch

+根据表达式显示或隐藏对应的部分

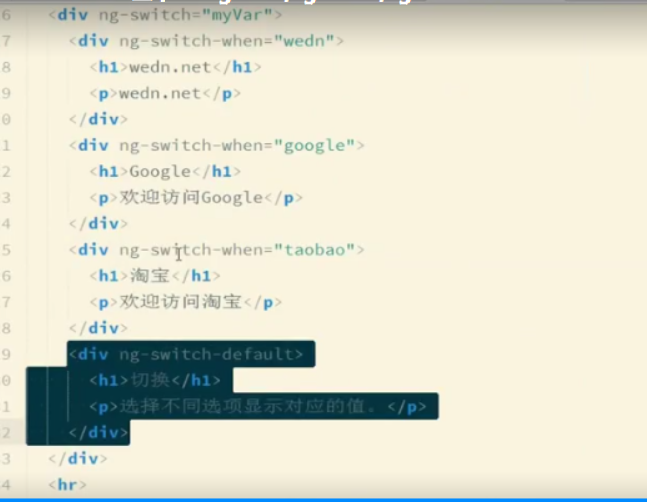
+对应的子元素使用ng-switch-when，如果匹配选中选择显示，其他为匹配的则移除

+你可以通过使用ng-switch-default设置默认选项，如果都没有匹配的情况，默认选项会显示

例子：选择网站，切换相应页面，没选则为默认状态







## 6.2.7 其他指令







以上选中事件可以用ng-module=“true”就可以

## 6.2.8 自定义指令（用的非常少）

目的：封装一些常用而且公用的东西，Angular仍然还有一些Dom操作的可能，所有的Dom操作都应该集中在自定义指令中

//第一个参数是指令的名字，第二个参数仍然应该使用一个数组，数组的最后一个元素是一个函数  
demoApp.**directive("itcastButton"**, **[function()**{}**])**;

# 7.DOM2

## 7.1可以绑定多个元素。传统：只能绑定一个事件

## 7.2动态的添加和移除事件

1）标准组织移除事件

removeEventListener：

-dom2级移除事件处理程序方法，参数及调用方式同addEventListener。

+btn.removeEventListener(“click”, Fn, false);

#### 注意：匿名函数不能移除

## 7.3 事件委托

定义：通俗的讲，事件就是onclick，onmouseover，onmouseout等，委托就是让别人来做，这个事件本来是加在某些元素上的，然而你却加到别人身上来做，完成这个时间。