1. Control负责把数据赋值给View:1)使用ViewBag来完成该任务。ViewBag是一个动态对象，可以给它的任意属性赋值。2)使用强类型视图。@model 权限定类.使用Model来在视图中引用强类型数据。
2. 加[HttpGet]或者[HttpPost]属性来使Action来只处理get或者post请求。
3. If语句： @if(){} else{} @switch(){} @foreach{}
4. 数据有效性验证，在System.ComponentModel.DataAnnotations命名空间，Required,RegularExpression注释。
5. Layout=null告诉MVC框架，视图是负责render全部内容。Layout负责给网站提供一致的界面，相当于MasterPage。在Views文件夹下以\_开头的文件，是不能被用户访问的。因此Layout文件应该是以\_开头的。@RenderBody()
6. Views\\_ViewStart.cshtml的内容就像它们被包含在视图的开始部分，（它先于视图执行,意味着可以在视图中覆盖——\_ViewStart的值）。即把视图的公共部分放到一个地方。
7. Razor表达式可以设置html属性。@:阻止Razor引擎将其翻译为C#语句。
8. @using在视图中引入命名空间。
9. Ninject是一个超轻量级，快速的DI以来注入容器。Entity是ORM tool.
10. 在Razor [view使用的类要么使用@using语句，要么在web.config](mailto:view使用的类要么使用@using语句，要么在web.config)声明。
11. Partial view是重用于嵌入其它view的。使用Html.Render Partial或者RenderAction来render partial view.
12. TempData temp data is deleted at the end of the HTTP request or it is readed.
13. routes.MapRoute(

name: "Default",

url: "{controller}/{action}/{id}",

defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }

);

routes.MapRoute("ShopSchema2", "Shop/OldAction", new { controller = "Home", action = "Index" });

routes.MapRoute("ShopSchema", "Shop/{action}", new { controller = "Home" });

routes.MapRoute("", "X{controller}/{action}");

routes.MapRoute("MyRoute", "{controller}/{action}", new { controller = "Home", action = "Index" });

routes.MapRoute("", "Public/{controller}/{action}", new { controller = "Home", action = "Index" });

Route并没有特殊的含义，是后面的mvc request handle给予了controller和action含义。