文章详细页的结构如下图：



首先到控制器PostController中新建actionDetail()动作：

分三部分内容：

*step1. 准备数据模型*

*step2. 当评论提交时，处理评论*

*step3. 传数据给视图渲染*

step2先留下，待完成详细页面后再做，具体见actionDetail。

文章详细页：

在view\post下新建一个detail.php，因详细页和文章首页index有很多相似的地方，因此可把首页的内容拷贝过来修改：



接下来展示文章内容：

有一点注意，由于用到了HTMLPurifier::process，我们在TestController中测试了HTMLPurifier::process与Html::encode的区别，从结果看：HTMLPurifier::process直接把js标签去掉了，Html::encode只是把HTML字符转换为Html实体，能显示出原有的标签，如script标签，但标签的内容无法运行，比如script标签中alert内容不会执行了。

<**div** class="content">  
 **<?=** HTMLPurifier::*process*($model->**content**);**?>**</**div**>

注意这里想起一个问题：就是前面把一些地方从Adminuser换为User的问题，这里是个人博客，所有文章理论上应该是博主也就是Adminuser发的，用户也就是User只能发表评论，所以视频讲的是没错的。

接下来，展示文章的标签，标签要可以点击：

<**div** class="nav">  
 <**span** class="glyphicon glyphicon-tag" aria-hidden="true"></**span**>  
 **<?=** implode(', ',$model->**tagLinks**);**?>** <**br**>  
 **<?=** Html::*a*("评论({$model->**commentCount**})",$model->**url**.'#comments');**?>** 最后修改于**<?=** *date*('Y-m-d H:i:s',$model->**update\_time**);**?>**</**div**>

下面的工作：



上图中提示框是可以关闭的，我们可以把bootstrap中可关闭的警告框例子代码拷贝过来修改。

注意$added表示是否有新的评论。后面会在控制器中定义。

接下来是这篇文章所有评论的展示框，先展示有多少条评论，具体评论内容要到\_comment视图展示，

再下来是发表评论部分，表单输入部分要到\_guestform视图生成。

$added还没有定义，我们需要到控制器PostController的actionDetail()中，来实现step2：

先添加一个成员变量或属性$added=0; 然后再编写step2的代码，完成后显示没问题。

下面编写\_guestform视图，我们可以借鉴后台的新增评论这个视图代码，拷贝过来修改：

用textarea做一个评论的输入框，详见页面程序。

整个前台页面显示没问题。