我们点击文章列表的show链接来查看一下文章的详情，这里出错了，它错误和之前的错误一样，没有找到post_edit路由。我们修改模板代码，回到项目，在templates目录下，我们打开show.html.twig文件，删除47行代码，我们看第48行代码，它使用了Twig方法include()将其他的模板文件引入到了当前的模板中。

我们不需要这个代码，你只需要对include()方法有个印象就可以了。在后面我们还会用到，删除这行代码，回到浏览器刷新，文章详情的地址是/post/6，我们回到PostController，第24行Route注解这里使用id占位符作为请求的属性参数，在show()方法中直接使用了$post对象，然后在render()方法中将$post对象渲染到了模板中，它是怎样从id参数转换为$post对象的呢?

很神奇!

我们修改一下id我们随便输一个字符，比如id1吧，回到浏览器再次刷新页面，这次报了个错误，我们查看一下错误栈。在查看之前我们回到项目，在config目录中，我们找到framework.yaml配置文件，我们修改一下配置文件，在framework配置键下能添加一下ide，这里设置为phpstorm。

再次回到浏览器刷新，配置ide设置后，我们可以在浏览器中快速的使用phpstorm定位到对应的类。在错误栈这里，我们找到了熟悉的handleRaw()方法，鼠标点击HttpKernel.php。我们点击后phpstorm会自动的打开这个类文件，我们按着command键进入handleRaw()方法。

在handleRaw()方法中，首先我们根据请求找到$controller，然后再设置$controller的参数，在第144行在获取参数之前，它会发送一个事件。事件系统我们在后面会继续讲解，然后会有对应的事件处理来处理这个事件。

我们回到浏览器，最后它找到了ParamConverterListener这个类来处理controller事件，我们点击这个类打开，在ParamConverterListener类中我们往下拉。这个类它监听了controller事件，然后使用onKernelController方法来对事件进行处理，再回到浏览器。最后它会调用DoctrineParamConverter类来对参数进行处理。

我们点击这个类，在这个类中我们找到apply()方法，在apply()方法第91行，我们看到了熟悉的find()方法，我们按着command键进入find()方法，在find()方法中首先获取EntityManager对象，然后获取对应的仓库，然后调用find()方法查找$id参数。如果没有找到对象，那么它将会尝试使用findOneBy()方法，按着command键进入findOneBy()方法。

在findOneBy()方法中，最终会使用Repository对象的findOneBy()方法来进行条件查询，找到我们的对象，再将对象设置到请求参数中，然后将对象通过依赖注入的方式设置到Controller的action方法中，我们直接就可以在action方法中使用这个对象了。

回到浏览器，搜索symfony paramconverter。我们查看Symfony官方的文档，这里有个ParamConverter注解，ParamConverter注解可以调用converter将请求的参数转换为对象，然后这个对象保存在请求的attributes属性中，然后可以将参数注入到Controller方法中，我们来验证一下这段话。

回到项目，在show()方法中添加Request对象，在方法体中我们查看Request对象，回到浏览器刷新，我们修改一下，这里修改为id。再刷新，我们展开attributes属性，在attributes属性中我们找到了post参数，它就是id为6的Post对象。

我们再回到项目，因为我们的请求参数id它是Post对象的一个属性，所以这里可以直接省略ParamConverter注解。如果这里的参数它不是Post类的属性，那么我们就需要手动添加ParamConverter注解了，修改id为id1添加注解。

ParamConverter的第一个参数就是我们的Post对象在attributes属性中的键，这里输入post，post要和依赖注入的参数名一致，然后我们对注解添加一些设置options，注解中我们让文章对象的id属性指向请求的id1参数，回到浏览器刷新，现在注解就起作用了。

```
class PostController extends AbstractController
{
    #[Route('/post/{id1}', name: 'post_show', methods: ['GET'])]
    #[ParamConverter('post', options: ['id' => 'id1'])]
    public function show(Request $request, Post $post): Response
    {
        return $this->render('post/show.html.twig', [
            'post' => $post,
        ]);
    }
}
```

我们删除第29行代码，再次刷新页面，文章现在就显示正常了，同样的我们要让封面图像显示为一张图片，我们修改模板代码，我们复制首页模板中的代码。粘贴，刷新，同样我们还需要修改正文，取消代码的转义。

现在文章就显示正常了，在下节课我们将在详情页上添加评论表单的功能。