继续查看doctrine生命周期事件的文档，我们往下看，我们可以创建一个单独的监听器或者订阅器类来增加代码的可复用性。这里有段监听器的示例代码，监听器类是个普通的PHP类，监听器类的方法就是生命周期的时间的回调方法。

示例代码是在对象更新前修改对象的数据，我们要设置子评论的文章属性，可以创建一个监听器类。然后在监听器的方法中获取到子评论对象，然后获取顶级的父评论对象，再将顶级父评论对象的post属性，设置到子评论的post属性中。

我们搜索`symfony entity listener`，打开Symfony文档，这里有段示例代码。我们手动创建的监听器类，要添加一个`doctrine.orm.entity_listener`标签，然后在类的注解中添加`EntityListener`注解。

回到代码，我们在src目录下新增一个目录，目录名称叫做Listener。在Listener中添加一个类，类名叫做CommentListener。回到浏览器查看doctrine文档。我继续往下看，这里有段示例代码，在类中添加了一些方法，方法前添加了doctrine事件注解。我们往下看，我们要在评论对象加载时对子评论对象进行处理，可以使用`postLoad`事件。

复制这个方法，回到项目粘贴，修改一下参数。第一个参数就是我们的Comment对象了，再引用一下参数的类型，引用一下注解，我们查看一下方法的参数。第一个参数是Comment对象，当doctrine加载数据库中的数据之后，会将数据库中的数据封装成一个对象。`postLoad`事件就发生在这个过程之后，我们获取了当前的评论对象，可以查看一下当前评论对象是否有父对象。

如果有父对象的话，我们获取顶级的父对象，再将顶级的父对象的post属性设置到当前的评论对象中。在类中我们新增一个方法，叫做`public function getParent()`，需要传一个参数，参数是当前的Comment对象，我们先来获取当前评论对象的父级。使用getParent()方法，如果当前的对象有父级评论，我们再次调用getParent()方法，getParent()方法参数就是父级评论对象，这是一个递归调用。

我们直接返回结果，如果当前对象没有复制评论的话，我们直接返回当前的评论对象。在postLoadHandle()方法中，我们使用getParent()方法来获取当前评论对象的顶级父评论对象，参数就是当前的评论对象，再将顶级父评论对象的post属性设置到$comment对象中。

```
#src/Listener/CommentListener.php

class CommentListener
{
    /** @ORM\PostLoad */
    public function postLoadHandler(Comment $comment, LifecycleEventArgs
$event)
    {
        $rootComment = $this->getParent($comment);
        $comment->setPost($rootComment->getPost());
    }

    public function getParent(Comment $comment)
    {
        $parent = $comment->getParent();
        if ($parent){
           return $this->getParent($parent);
        }
```

```
            return $comment;
        }
    }
```

现在CommentListener类就已经编写好了，我们可以通过注解的方式来使用这个类。打开Comment类，在Comment类前我们添加注解EventListeners，注解的名称是个复数，这里需要传递一个数组，使用大括号。我们需要传入Listener全类名，回到CommentListener。我们复制类名，粘贴。

回到管理端，我们看到id为109和107的评论，文章标题这里显示的未赋值。现在我们刷新列表，现在就显示了文章的标题，我们编写的监听器类就生效了。我们没有为CommentListener类加tags标签。这要感谢Symfony的自动配置功能，它自动的为CommentListener类添加了标签。

回到Comment类，我们查看一下$children属性，$children属性这里我们设置了级联删除。当删除评论对象时，也会把所有的子评的对象一块删除，我们可以在CommentListener类中进行处理。在删除评论对象时，如果当前评论对象有子评对象，我们把当前评论对象的父级设置到子级评论对象上，然后修改子级评论对象的$level属性。再删除当前的评论对象，这样就可以只删除父级评论对象，而不删除子级评论对象。

在下节课我们来继续修改CommentListener类。