我们打开PostCrudController类，第16行有个注释掉的方法configureFields()方法，configureFelds()方法用来对Post类的各个属性来设置字段的类型，现在它被注释掉了，所以会调用父类AbstractCrudController类的configureFields()方法，来自动的为Post类的各个属性设置字段的类型。

按着command键点击AbstractCrudController类，在configureFileds()方法中，EasyAdminBundle会读取Post类的各个属性的ORM注解，然后根据注解的类型设置不同的字段类型，我们查看EasyAdminBundle的文档。

我们查看Fields文档，EasyAdminBundle提供了一些内置的字段类型，比如说ID字段，当EasyAdminBundle读取Post类的所有属性的注解时，发现$id属性它的类型是Id类型，那么就会自动的为$id属性设置为IdField字段类型。再举例，它查看到$title属性的注解类型是string类型，那么它会自动的为$title属性设置为TextField类型。

我们可以在PostCrudController类中手动的设置Post类各个属性的字段类型，我们取消configureFields()方法的注释，id属性就是IdField类型，我们直接使用IdField。title属性是string类型，所以使用TextField。下面是摘要属性，可以使用TextAreaField。下面是正文属性，正文属性我们希望能够使用到HTML文本编辑器，EasyAdminBundle内置了一个文本编辑器，它的字段类型是TextEditorFileld。我们使用new方法输入body，下面是$status属性。$status属性我们想通过使用下拉选项的方式来选择文章的状态，输入ChoiceField。属性名是status，我们还需要为下拉框添加选项，我们按着command键点击ChoiceField查看源码。在ChoiceField中有个setChoices()方法，可以为下拉框添加选项，查看行前的注释，这里有个示例代码，我们直接复制66行的示例代码。

回到PostCrudController类，在下一行直接粘贴代码，这样setChoices()方法就为ChoiceField添加了两个选项，第一个选项的名称是foo，value值是1，第二个选项它的名称是bar，它的选项值为2。

我们修改一下，修改第一个选项，它的名称为draft，草稿状态，它的值也为draft。第二个选项为published，选项值也为published。 我们可以查看一下页面的效果，回到浏览器，我们点击Add Post按钮，页面上显示的字段就是刚刚我们配置的字段，我们来看status字段。我们刚刚设置了两个选项，就已经有了。

回到项目，继续设置字段，下面属性是createdAt、updatedAt，它们的类型是datetime，我们可以使用TimeField字段。输入TimeField::new()，属性名输入createdAt，我们按着command键查看TimeField源码，在TimeField源码中可以设置时区，设置日期的显示格式。

回到controller，我们设置一下日期的显示格式，格式的字符串，这里直接使用最常用的年月日时分秒。复制TimeField这行，然后新增一行，属性名称我们设置为updatedAt。

我们查看Post类，最后我们需要上传一张图片作为封面图像，EasyAdminBundle为我们提供了图像字段，用于图像的上传，我们回到Controller，输入ImageField，属性名就是postImage。

同样按着command键查看ImageField源码，我们需要设置图像的basePath和文件上传目录。我们在public目录中创建一个文件夹，我们将所有的封面图像都保存到images这个文件夹下。回到controller，设置basePath为uploads/images。然后设置上传文件的目录，basePath目录用于图片的显示，当访问图片链接时，会在图像名称前自动添加这个路径，我们还可以设置文件上传后保存的文件名。

我们打开ImageField源码，在最下方有个setUploadedFileNamePattern()方法，方法前的注释，第63行我们可以通过表达式的方式来设置文件保存的名称，我们复制63行代码，我们保存文件的别名和哈希值以及文件的后缀。

回到Controller类，回到浏览器，我们点击添加文章，我们发现了一些问题，文章的ID是在插入数据库时自动生成的，我们不需要在表单中显示这个表单行。文章的添加时间和更新时间要自动生成的，也不需要在这里生成表单行。修改controller代码，Field类提供了另外一个方法`onlyOnIndex()`。这个方法让ID字段只在首页显示，同样的方法我们设置createdAt和updatedAt字段，再次刷新页面，我们还希望让封面图像显示在标题下方，我们可以调整字段的顺序。

刷新，现在我们来手动创建一篇文章，文章的状态，我们选择为草稿状态点击Create按钮。这里提示出错了，图像在上传时不能猜测图像文件的后缀，我们需要安装`mime`组件。

```
class PostCrudController extends AbstractCrudController
{
    // ...

    public function configureFields(string $pageName): iterable
    {
        return [
            IdField::new('id'),
            IdField::new('id')->onlyOnIndex(),
            TextField::new('title'),
            TextEditorField::new('description'),
            ImageField::new('postImage')
                ->setBasePath('uploads/images')
                ->setUploadDir('public/uploads/images')
                ->setUploadedFileNamePattern('[slug]-[contenthash].
[extension]'),
            TextareaField::new('summary'),
            TextEditorField::new('body'),
            ChoiceField::new('status')
                ->setChoices(fn() => ['draft' => 'draft', 'published' =>
'published']),
            TimeField::new('createdAt')
                ->setFormat('Y-MM-dd HH:mm:ss')->onlyOnIndex(),
            TimeField::new('updatedAt')
                ->setFormat('Y-MM-dd HH:mm:ss')->onlyOnIndex(),
        ];
    }

    // ...
}
```

复制这个命令行，回到项目，打开控制台，粘贴。

在安装成功后，我们点击刷新按钮再次提交表单，这是我们手动添加的文章，就成功的创建了。但是现在有一个问题，我们想让最新添加的文章显示在前面，ID按降序的方式进行排列。

回到Controller类，CrudController类的父类提供了另外一个方法，叫做`configureCrud()`。我们可以使用Crud的`setDefaultSort()`方法，来对列表的数据进行排序。覆盖`configureCrud()`方法，使用Crud的`setDefaultSort()`方法来对列表的结果进行排序，我们让ID按照降序的方式进行排列，最后返回一下。

```
class PostCrudController extends AbstractCrudController
{
    // ...

    public function configureCrud(Crud $crud): Crud
    {
        return $crud->setDefaultSort(['id' => 'DESC'])
            ->setSearchFields(['title', 'summary', 'body']);
    }

    // ...
}
```

刷新文章列表页，这样最新创建的文章就第一个显示了，在文章列表页的上方有个搜索框，默认情况下，它会搜索文章类的所有属性，比如说我们搜索draft，它会搜索文章的所有属性。如果属性中包含draft字符串，那么它会在列表中进行显示。我们搜索draft回车，状态为draft的文章都显示了出来，现在我们想让搜索框只搜索标题、摘要和正文。

回到项目，我们可以对Crud对象进行设置，添加代码，通过setSearchFields()方法，我们可以对搜索框进行字段的限制，现在我只搜索标题摘要和正文，我们再次进行搜索。现在就收不到结果了，但是如果我们搜索Post，它将会搜索出所有标题摘要和正文，包含Post字符串的结果。

我们还可以对列表添加一个过滤器，回到项目，AbstractCurdController类中有一个configureFilters()方法。我们可以设置$filters，回到PostCrudController，覆盖configureFilters()方法，我们可以使用$filters对象的add()方法来添加过滤器，我想对文章的状态进行过滤，可以使用ChoiceFilter，我们选择EasyAdminBundle的ChoiceFilter，new()方法的参数是属性名称，这里输入status，我们可以使用setChoices()方法设置过滤选项。

setChoices()的参数是个数组，这里我们输入draft，published，值也为published。数组的键用作下拉选项的标签，数组的值用作下拉选项的value，我们回到页面刷新。

```
class PostCrudController extends AbstractCrudController
{
    // ...

    public function configureFilters(Filters $filters): Filters
    {
        return $filters->add(ChoiceFilter::new('status')
            ->setChoices(['draft' => 'draft', 'published' =>
'published']));
    }

    // ...
}
```

页面上增加了一个过滤器按钮，我们点开这个按钮勾选status，我们过滤草稿状态的文章，点击Apply，结果就是草稿状态的文章。我们也可以过滤出不是草稿状态的文章，选择is not same，点击应用。现在我们就过滤出了所有published状态的文章，我们使用EasyAdminBundle快速的创建了Post类的管理页面。

在下一节课，我们将继续使用EasyAdminBundle来创建Comment类的管理页面。