在Teebblog项目中我们使用FileManaged类封装文件信息，现在我们需要在Bundle中创建一个类，将文件信息封装一下。在src目录下创建Entity目录，手动创建一个类名称叫作File吧，并且添加一些属性。

```
class File
{
    private $id;

    private $originName;

    private $name;

    private $size;

    private $mimeType;

    /**
     * @return int
     */
    public function getId()
    {
        return $this->id;
    }

    /**
     * @return string
     */
    public function getOriginName()
    {
        return $this->originName;
    }

    /**
     * @param string $originName
     */
    public function setOriginName($originName): void
    {
        $this->originName = $originName;
    }

    /**
     * @return string
     */
    public function getName()
    {
        return $this->name;
    }

    /**
     * @param string $name
     */
    public function setName($name): void
    {
```

```
        $this->name = $name;
    }

    /**
     * @return int
     */
    public function getSize()
    {
        return $this->size;
    }

    /**
     * @param int $size
     */
    public function setSize($size): void
    {
        $this->size = $size;
    }

    /**
     * @return string
     */
    public function getMimeType()
    {
        return $this->mimeType;
    }

    /**
     * @param string $mimeType
     */
    public function setMimeType($mimeType): void
    {
        $this->mimeType = $mimeType;
    }

}
```

File类中的数据最终是要保存到数据库的，我们需要添加doctrine的ORM映射信息。为了兼容性这里还是使用xml配置文件进行映射吧。下节课，我会讲解使用注解的方式。我更喜欢注解的方式，但是为了兼容性，先来讲一下xml的方式。我们的数据库有orm类型，odm类型，这里只讲解orm类型的配置。

写注解写多了 对xml配置我也并不熟悉，但是我会ctrl c/v，搜索`doctrine xml mapping`, 直接复制，粘贴，修改。

```
<?xml version="1.0" encoding="UTF-8"?>
<doctrine-mapping xmlns="https://doctrine-
project.org/schemas/orm/doctrine-mapping"
                  xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance"
                  xsi:schemaLocation="https://doctrine-
project.org/schemas/orm/doctrine-mapping
                          https://www.doctrine-
```

```
project.org/schemas/orm/doctrine-mapping.xsd">

    <entity name="Teebb\UploadBundle\Entity\File" table="file">

        <id name="id" type="integer" column="id">
            <generator strategy="AUTO"/>
        </id>

        <field name="originName" column="origin_name" type="string"
length="255"/>
        <field name="name" column="name" type="string" length="255"/>
        <field name="size" column="size" type="integer"/>
        <field name="mimeType" column="mime_type" type="string"
length="100"/>

    </entity>

</doctrine-mapping>
```

和之前的xml配置一样，我们也需要让配置文件起作用。

这个xml文件是让doctrine组件读取的，我们需要操作Doctrine，让他读这个配置文件。这就不能在`TeebbUploadExtension`类中操作了。我们需要修改`TeebbUploadBundle`类，在Symfony准备容器时，会读取所有Bundle然后进行一些设置。

覆盖`TeebbUploadBundle`父类的`build`方法。这里额外讲一下Symfony的原理：当我们写好代码，启动项目，访问第一个页面时，Symfony首先会准备容器，容器其实是一个类，Symfony会把所有的Bundle服务类、我们自己写的服务类、配置信息、都添加到一个完整的容器里。在添加时，会进行一些处理，比如服务类中构造方法参数的设置等等。

这个过程可以叫做`CompilerPass`。 Doctrine提供了一个`CompilerPass`类，在将Doctrine中的服务类添加到容器时，通过`CompilerPass`类，我们可以让Doctrine读取我们自己的xml配置文件。

我们的Bundle需要用到doctrine，安装一下这个包，可以在开发时自动进行提醒。`composer req symfony/orm-pack`，如果安装出错，则是`orm-pack`中有已安装的包的依赖，需要先卸载出错的组件，再重新安装。

```
class TeebbUploadBundle extends Bundle
{
    public function build(ContainerBuilder $container)
    {
        parent::build($container);

        // 此处映射的是类目录对应的namespace
        $mappings = [
            realpath(__DIR__.'/Resources/config/doctrine-mapping') =>
'Teebb\UploadBundle\Entity',
        ];

        //此处可以添加验证，DoctrineOrmMappingsPass类是否存在
        if (class_exists(DoctrineOrmMappingsPass::class)) {
```

```
            $container-
>addCompilerPass(DoctrineOrmMappingsPass::createXmlMappingDriver($mappings
));
        }
    }
}
```

最后在项目中使用命令行`symfony console doctrine:mapping:info`查看我们的类是否已经映射成功了。

```
 Found 5 mapped entities:

 [OK]   App\Entity\Comment
 [OK]   App\Entity\Post
 [OK]   App\Entity\User
 [OK]   App\Entity\FileManaged
 [OK]   Teebb\UploadBundle\Entity\File
```

我们可以使用命令行创建表了。`symfony console make:migration`查看migration文件，我们的映射已经成功起作用了。

下一节我们使用注解来创建Entity类的映射。