错误提示我们有两种方式解决，上节课我们已经看了FileType的源码，data_class是Symfony提供的File类型，当使用FileType类型的表单行时，Symfony会自动将数据转换为File类型，UploadedFile类型是File类型的子类。

我们可以在FileManagedType类中手动将data_class设置为null

```
    public function configureOptions(OptionsResolver $resolver)
    {
        $resolver->setDefaults([
            'data_class' => null,
        ]);
        // setDefined定义一个option名称，这样我们可以在创建表单行时为这个名称设置
值。
        // 为了防止漏用这个option，添加另外一个设置 setRequired ，这
样'file_class'选项就是必填项。
        // 使用setDefaults设置选项的默认值
        $resolver
            ->setDefined('file_class')
            ->setRequired('file_class');
    }
```

再回到评论编辑页面，表单显示的就正常了。

当前的评论表单有一个文件上传input行，这就说明评论对象中有一个文件。如果有多个文件会显示多行，后面我们会修改代码让文件已超链接或者图片的方式显示出来。

修改表单提交事件的代码： 有三种情况： 当新增评论时，评论表单中是没有数据的，如果上传了文件，将会new一个文件对象，我们之前的代码就是这么做的，功能也都没有问题。

第二种情况： 当编辑评论表单时，编辑了表单中的其他行，没有编辑文件行，这时，我们要获取到表单中文件的数据，再设置回表单中。

第三种情况： 当编辑评论表单时，编辑了文件行，这时我们要修改表单中旧的文件数据，替换为新的文件数据。

修改代码：

```
    public function buildForm(FormBuilderInterface $builder, array
$options)
    {
        $fileClass = $options['file_class'];
//        $builder->addEventListener(FormEvents::SUBMIT,[$this,
'onFormSubmit']);

        $builder->addEventListener(FormEvents::SUBMIT, function (FormEvent
$event) use ($fileClass) {
```

```
            /**@var UploadedFile|null $data * */
            //这里获取到的是表单提交时的数据
            $file = $event->getData();

            /**@var File|null $formFileObject * */
            //这里获取到的是表单行里之前的数据
            $formFileObject = $event->getForm()->getData();

            if ($file && $file instanceof UploadedFile &&
!$formFileObject) {
                //如果表单行中没有数据且新上传了文件，那么就new一个新的文件对象并设置
属性的值，

                $formFileObject = new $fileClass();

                //提取共同的代码
                $this->fillFileObject($file, $formFileObject);

            } elseif ($file && $file instanceof UploadedFile &&
$formFileObject) {
                //如果表单行中有数据且新上传了文件，那么就更新旧的文件对象，

                //提取共同的代码
                $this->fillFileObject($file, $formFileObject);

            }

            $fileUploadEvent = new AfterFileUploadEvent($file,
$formFileObject);
            $this->eventDispatcher->dispatch($fileUploadEvent);

            //这里把数据设置回$event 进行后续处理。
            $event->setData($fileUploadEvent->getFileObject());

        });

    }

    private function fillFileObject(UploadedFile $file, File $fileObject)
    {
        $originName = $file->getClientOriginalName();
        $fileName = pathinfo(htmlspecialchars($originName),
PATHINFO_FILENAME) . '-' . $file->getFilename() . '.' . $file-
>getClientOriginalExtension();

        $mimeType = $file->getMimeType();
        $filesize = $file->getSize();

        $file->move($this->uploadDir, $fileName);

        $fileObject->setName($fileName);
        $fileObject->setMimeType($mimeType);
        $fileObject->setOriginName($originName);
```

```
            $fileObject->setSize($filesize);
        }
```

修改AfterFileUploadEvent类，UploadedFile对象有可能为null值。

```
class AfterFileUploadEvent extends Event
{
    /**
     * @var UploadedFile|null
     */
    private $uploadedFile;
    /**
     * @var File
     */
    private $fileObject;

    public function __construct(?UploadedFile $uploadedFile, File
$fileObject)
    {
        $this->uploadedFile = $uploadedFile;
        $this->fileObject = $fileObject;
    }

    /**
     * @return UploadedFile|null
     */
    public function getUploadedFile(): ?UploadedFile
    {
        return $this->uploadedFile;
    }

    /**
     * @return File
     */
    public function getFileObject(): File
    {
        return $this->fileObject;
    }

}
```

现在文件表单上传的功能基本已经做完了，有几个问题：

1. 文件表单行的回显， 显示一个空白的表单行不行，如果是编辑表单，那么就显示一个文件链接或者一个img标签把图片显示出来。
2. 我们的bundle是要可复用的，灵活性上可以更强一些，比如上传后文件的命名规则，是不是可以自定义呢?
3. 文件修改后，需不需要删除旧文件？虽然现在磁盘空间不贵了，但是有些情况下，还是可以删除一下的。

4. 目前我们所有的文件操作都在本地，如果将来，需要把文件移动到第三方云平台，那么直接使用move()这个方法就不行了。

后面的课程我们会解决这些问题。下节课，我们先把表单行的样式进行修改。