bundle的配置文件recipe

测试一下文件上传！！！页面出错了，但是文件成功上传到了目录中。

FileManagedType类中文件上传的目录是在`service.yaml`配置文件中配置的。我们希望文件上传目录在Bundle的配置文件中进行配置。

Recipe就是Bundle的配置文件，配置文件通常用yaml格式编写。

查看其他的Bundle源码，在`DependencyInjection`目录下有`Configuration`类，要实现`ConfigurationInterface`接口，这个接口是Symfony的Config组件提供的，我们安装一下组件，这样就不会出现找不到类的错误提示了。

```
composer require symfony/config
```

看类的代码，配置文件的根结点名称是通过 TreeBuilder 类生成的。

添加我们自己的Configuration类：

```
class Configuration implements ConfigurationInterface
{

    public function getConfigTreeBuilder()
    {
        $builder = new TreeBuilder('teebb_upload');
        #此处为了兼容旧版本的Symfony
        if (\method_exists($builder, 'getRootNode')) {
            $rootNode = $builder->getRootNode();
        } else {
            // BC layer for symfony/config 4.1 and older
            $rootNode = $builder->root('teebb_upload');
        }

        $rootNode
            ->children()
            ->scalarNode('upload_dir')->isRequired()->cannotBeEmpty()-
>end()
        ;

        return $builder;
    }
}
```

在`teebb_upload`根结点下添加了`upload_dir`配置项，我们可以在项目的config/packages目录下添加一个配置文件`teebb_upload.yaml`。yaml文件的文件名并不重要，但是配置文件里的根结点一定要是我们在Configuration类中定义的。

```
#config/packages/teebb_upload.yaml

teebb_upload:
    upload_dir: '%kernel.project_dir%/public/teebb_upload'  #需要在项目
public目录下添加teebb_upload目录
```

配置文件的作用就是让bundle获取里面配置的参数，然后完成不同的功能，我们的配置文件，Bundle现在并没有进行读取。

在TeebbUploadExtension类中，我们参考其他bundle中的代码进行编写。

```
class TeebbUploadExtension extends Extension
{

    public function load(array $configs, ContainerBuilder $container)
    {
        $processor = new Processor();
        $configuration = new Configuration();
        $config = $processor->processConfiguration($configuration,
$configs);

dd($config);

        $loader = new XmlFileLoader($container, new
FileLocator(__DIR__.'/../Resources/config'));
        $loader->load('form.xml');

    }
}
```

我们需要在FileManagedType类中处理上传的文件时用到这个变量，第一种方法：我们可以把这upload_dir以参数的形式添加到容器中。再到FileManagedType获取这个变量使用，这样在其他地方如果仍然需要这个参数的话也可以直接获取。

```
class TeebbUploadExtension extends Extension
{

    public function load(array $configs, ContainerBuilder $container)
    {
        $processor = new Processor();
        $configuration = new Configuration();
        $config = $processor->processConfiguration($configuration,
$configs);

        $loader = new XmlFileLoader($container, new
FileLocator(__DIR__.'/../Resources/config'));
        $loader->load('form.xml');
```

```
        #这样处理
        $container->setParameter('teebb.upload_dir',
$config['upload_dir']);
    }
}
```

再回到FileManagedType类中，使用ParameterInterface接口获取参数。

```
#FileManagedType.php
class FileManagedType extends AbstractType
{
    /**
     * @var ParameterBagInterface
     */
    private $parameterBag;

    public function __construct(ParameterBagInterface $parameterBag)
    {
        $this->parameterBag = $parameterBag;
    }

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

    }

    public function onFormSubmit(FormEvent $event){
        /**@var UploadedFile $data**/
        $file = $event->getData();
        $originName = $file->getClientOriginalName();
        $fileName = pathinfo(htmlspecialchars($originName),
PATHINFO_FILENAME) . '-' . $file->getFilename() . '.' . $file-
>getClientOriginalExtension();
        $uploadPath = $this->parameterBag-
>get('teebb.upload_dir');//$this->getParameter('base_path');

dd($uploadPath);

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

        $file->move($uploadPath, $fileName);

    }

    public function getParent()
    {
        return FileType::class;
    }
```

```
}
```

上传一个文件测试一下。 我们自己的配置成功应用了。

下一节，我们修改FileMangedType类的构造方法，直接把`upload_dir`配置项设置为构造函数的参数，也就是构造方法的参数注入。