搜索symfony workflow，我们查看下面这个文档。首先我们要安装workflow组件，复制命令行，粘贴。

```
composer require symfony/workflow
```

组件安装完成后，我们打开config目录。在config目录中，flex组件为我们添加了workflow.yaml配置文件，我们需要在workflows配置下添加自定义的工作流配置。

回到浏览器，我们继续往下看，这里有段工作流配置的案例。blog_publishing是工作流的名称，type是工作流的类型。工作流只有两种类型，第一种就是工作流，第二种就是状态机。

对于文章发布，我们使用工作流，下面是添加审计信息，这里设置为true，会自动的为工作流添加一些日志信息。marking_store配置下type配置我们只能使用method。

然后是属性，我们指定类中的哪个属性来保存文章的状态，这里使用了currentPlace。suppoorts配置下是需要使用工作流的Entity类。第13行初始化的marking就是初始化时文章的状态，第14行places下配置的是文章所有可用的状态，第19行配置的是文章的所有状态的转移。

to_review是状态转移的名称，from和to配置的是从哪个状态转移到哪个状态。我们可以参考这段配置来编写我们自己的配置。回到项目，我们在workflows下配置自己的工作流。

工作流的名称，我们叫做blog_publishing，首先我们指定工作流的类型，复制第5行代码，粘贴。然后我们开启审计信息，复制第8行到第10行代码，我们指定一个属性用来保存当前工作流的状态。打开Post类，在Post类中，我们定义了$status属性用来保存当前的文章状态。这里我们修改为status。

复制supports配置项，这里我们的类修改为Post类，然后我们指定初始的状态，复制13行到18行代码，粘贴。初始的状态我们也定义为草稿状态，我们需要对齐places设置。在places下，我们定义文章所有的状态，首先是草稿状态，然后是wait_for_review，wait_for_check， 然后是review成功状态，然后是拼写检查成功状态，最后是发布成功状态。

下面我们定义一些状态的转移transitions，粘贴。首先我们定义review_request，from从草稿状态到wait_for_review，wait_for_check。然后我们定义editor_review，from: wait_for_review，to: approved_byeditor。然后我们定义checker_check，拼写检查from: wait_for_check，to: approved_by_checker。然后我们定义publish，from只有当文章处于这两个状态时，我们的文章才允许发布。粘贴to: published。

```
#config/packages/workflow.yaml

framework:
    workflows:
        blog_publishing:
            type: 'workflow' # or 'state_machine'
            audit_trail:
                enabled: true
            marking_store:
                type: 'method'
                property: 'status'
            supports:
                - App\Entity\Post
```

```
            initial_marking: draft
            places:
                - draft
                - wait_for_review
                - wait_for_check
                - approved_by_editor
                - approved_by_checker
                - published
            transitions:
                review_request:
                    from: draft
                    to: [wait_for_review, wait_for_check]
                editor_review:
                    from: wait_for_review
                    to: approved_by_editor
                checker_check:
                    from: wait_for_check
                    to: approved_by_checker
                publish:
                    from: [approved_by_editor, approved_by_checker]
                    to: published
```

现在我们就定义好了文章发布的工作流，我们可以使用命令行生成当前工作流的图表。回到文档，我们继续往下看，查看这个文档。我们可以使用这个命令行来为工作流生成图表。首先需要安装对应的库，我这里已经安装了Graphviz。我们将使用这个库来生成当前的工作流图表。复制第一个命令行，回到项目，打开控制台粘贴。

修改一下，这里我们要修改一下工作流的名称，修改为blog_publishing。文件的名称，我们修改为blog_publishing.svg文件。

```
php bin/console workflow:dump blog_publishing | dot -Tsvg -o
blog_publishing.svg
```

执行命令行，我们查看项目根目录，在根目录中生成了点SVG文件。我们查看一下文件，我们文章发布之后，首先是草稿状态，然后等待复查的请求，应用完复查请求之后等待编辑的复查，编辑审核之后会转移到approved_by_editor状态。同样的拼写检查之后会转移到approved_by_checker这个状态。当文章处于这两个状态时，就可以执行发布转移，最终文章就会转换为published状态。和之前我们查看到的案例的流程是一样的。

在下节课我们对Post类进行修改，让$status属性可以同时保存多个状态。