回到浏览器，我们搜索knppaginatorbundle，打开github这个链接，这里有这个包的说明。它是一个Symfony的分页器，可以对结果进行排序和分页。我们向下看，这里有安装方法：

```
composer require knplabs/knp-paginator-bundle
```

我们复制命令行安装一下，回到项目打开控制台粘贴命令行，再回到浏览器，我们查看一下使用的方法。首先对这个包进行一些配置，然后往下看，我们找到了一个controller方法。

在controller方法中，首先通过依赖注入来注入一个PaginatorInterface对象，然后使用$paginator的paginate()方法来对查询进行分页。

它的第一个参数是Query对象，第二个参数是当前的页码，第三个参数是当前页面显示数量。然后将$pagination对象渲染到模板中，在模板中可以通过pagination.getTotalItemCount来统计所有的结果的数量。然后在表单体中，使用for循环来遍历pagination中的所有结果，最后使用Twig方法来渲染分页器的页码。

回到项目，我们需要在文章的详情页对评论列表进行分页，打开PostController，在show()方法中，我们依赖注入$paginator对象，PaginatorInterface类型，回到浏览器，我们复制代码，粘贴。现在$paginator的paginate()方法缺少一个Query对象，Query对象我们还通过Repository类的一个方法来获取，这次我们使用CommentRepository类，打开CommentRepository添加一个方法getPaginationQuery()。

方法的返回值是Query对象，我们依然使用$this->createQueryBuilder()方法，来创建一个QueryBuilder对象，然后使用QueryBuilder对象的getQuery()方法来获取一个Query对象。

引入一下Query的类型，我们需要对评论进行一下查询，添加一个查询条件，andWhere()，我们需要查询某个文章下的所有评论。设置文章参数，文章对象我们依然通过方法的参数来获取，我们还要对结果进行排序，orderBy()，c.id我们按照降序进行排序，这里引入一下Post类。

```
#src/Repository/CommentRepository.php

class CommentRepository extends ServiceEntityRepository
{
    public function getPaginationQuery(Post $post):Query
    {
        return $this->createQueryBuilder('c')
            ->andWhere('c.post = :post')
            ->setParameter('post', $post)
            ->orderBy('c.id', 'DESC')
            ->getQuery();
    }
}
```

回到Controller，我们在参数中依赖注入CommentRepository类对象。换下行，我们使用CommentRepository类的getPaginationQuery()方法来返回一个Query对象，参数就是文章对象，现在我们将$pagination对象渲染到模板中。回到模板，我们修改一下评论列表，这里就要遍历pagination对象了。

```
#src/Controller/PostController.php
class PostController extends AbstractController
{
    // ...
    #[Route('/post/{id1}', name: 'post_show', methods: ['GET', 'POST'])]
    #[ParamConverter('post', options: ['id' => 'id1'])]
    public function show(Request $request, Post $post,
EntityManagerInterface $entityManager,
                         PaginatorInterface $paginator, CommentRepository
$commentRepository): Response
    {
        $commentForm = $this->createForm(CommentType::class);

        $commentForm->handleRequest($request);
        if ($commentForm->isSubmitted() && $commentForm->isValid()) {
            if ($commentForm->get('submit')->isClicked()) {
                /**@var Comment $data * */
                $data = $commentForm->getData();
                $data->setPost($post);
                $entityManager->persist($data);
                $entityManager->flush();
            }
        }

        $query = $commentRepository->getPaginationQuery($post);
        $pagination = $paginator->paginate(
            $query, /* query NOT result */
            $request->query->getInt('page', 1), /*page number*/
            10 /*limit per page*/
        );

        return $this->render('post/show.html.twig', [
            'post' => $post,
            'pagination' => $pagination,
            'comment_form' => $commentForm->createView()
        ]);
    }
    // ...
}
```

回到浏览器，在模板中，我们还需要渲染一个页码器，我们复制这四行代码。回到项目，在for循环下面我们直接粘贴代码。打开博客首页，打开最新的文章，现在文章下面的评论列表就显示了10条(顶级)评论，在评论列表下方是页码，但是样式显示的很简单。

我们再看配置PaginatorBundle的README文件，我们往上看，我们可以对KnpPaginatorBundle进行一些设置，比如说设置每一页显示的数量，然后还可以设置页码参数的名称。在template键下，我们还可以设置分页器显示模板，我们复制所有配置。

回到项目，在config目录下打开packages目录，我们新添加一个yaml文件，文件名为叫做 `knp_paginator.yaml`（配置文件的文件名不重要，仅用于配置文件的管理），粘贴代码。

```
#config/packages/knp_paginator.yaml
knp_paginator:
    page_range: 5                       # number of links showed in the
pagination menu (e.g: you have 10 pages, a page_range of 3, on the 5th
page you'll see links to page 4, 5, 6)
    default_options:
        page_name: page                 # page query parameter name
        sort_field_name: sort           # sort field query parameter name
        sort_direction_name: direction  # sort direction query parameter
name
        distinct: true                  # ensure distinct results, useful
when ORM queries are using GROUP BY statements
        filter_field_name: filterField  # filter field query parameter
name
        filter_value_name: filterValue  # filter value query parameter
name
    template:
        pagination:
'@KnpPaginator/Pagination/twitter_bootstrap_v4_pagination.html.twig'     #
sliding pagination controls template
        sortable: '@KnpPaginator/Pagination/sortable_link.html.twig' #
sort link template
        filtration: '@KnpPaginator/Pagination/filtration.html.twig'  #
filters template
```

我们按着command键点击pagination这个模板文件，我们打开模板文件所在的目录。双击，在模板目录中，我们找到了boostrap页码器模板，我们复制文件名。回到配置文件修改一下，再回到浏览器刷新一下文章详情页，没有起作用。

回到项目，这是因为我们在config目录下新增了一个配置文件，Symfony没有读取这个新的配置文件，我们需要清理一下缓存。输入`symfony console cache:clear`，缓存清理成功后再次刷新详情页，现在在评论列表下方就显示了新的页码器。

我们看到页码器上一页按钮和下一页按钮是英文的，在下节课我们将页码器的按钮进行一下国际化的翻译。