ASP.NET Core 基础系列（9）(异常中间件)

原创 痕迹 CodeShare 今天

收录于话题
#ASP.NET Core

9个

了解异常中间件

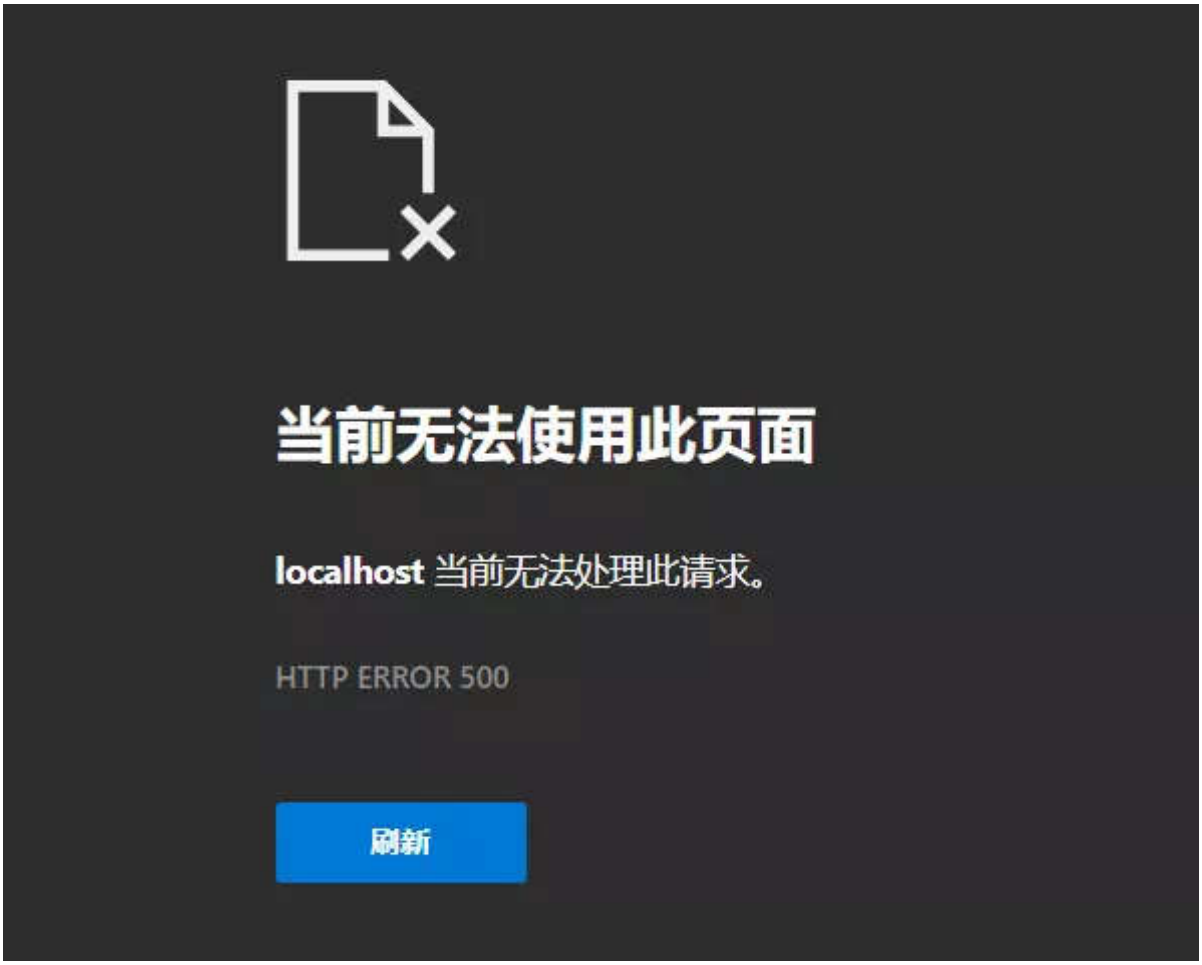
首先，使用ASP.NET模板创建一个核心应用程序。默认情况下，ASP.NET核心应用程序只是返回应用程序未处理的异常的状态代码。如下所示，我们引发异常。

```
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{

    app.UseRouting();

    app.UseEndpoints(endpoints =>
    {
        endpoints.MapGet("/", async context =>
        {
            throw new Exception("****");
            await context.Response.WriteAsync("Hello World!");
        });
    });
}
```

运行应用程序时，将得到以下输出。



如上图所示，它为您提供的状态代码为 500，这意味着内部服务器错误。但是，作为开发人员，在开发应用程序时，您应该知道有关页面上异常的详细信息，以便可以采取必要的操作来修复错误。

如何使用异常中间件?

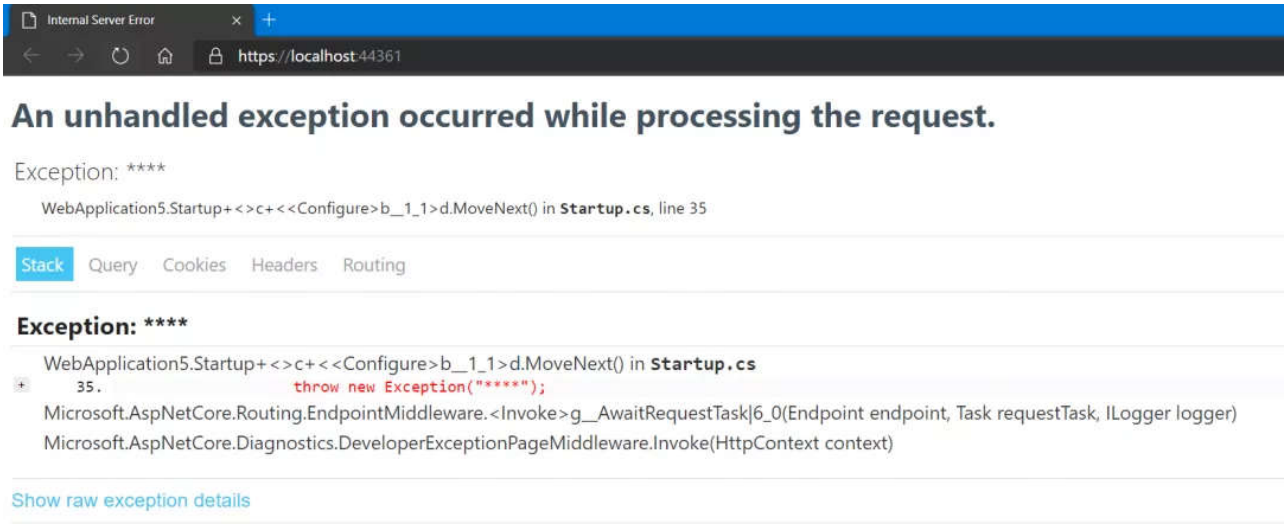
如果希望应用程序显示显示有关未处理异常的详细信息的页面，则需要在请求处理管道中配置开发人员异常页面中间件。
为此，请修改Startup类的 Configure 方法，如下所示，以添加开发人员异常页中间件，该中间件将处理应用程序中发生的未处理异常。

```
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }

    app.UseRouting();

    app.UseEndpoints(endpoints =>
    {
        endpoints.MapGet("/", async context =>
        {
            throw new Exception("****");
            await context.Response.WriteAsync("Hello World!");
        });
    });
}
```

现在运行该应用程序，它将显示以下页面，其中包含有关未处理异常的详细信息。



如上图所示，“开发人员异常”页面包含五个选项卡，例如“堆栈”，“队列”，“Cookie”，“标题”和“路由”。

1.堆栈：“堆栈”选项卡提供堆栈跟踪信息，该信息指示确切的异常发生位置，文件名以及导致异常的行号。

2.查询：“查询”选项卡提供有关查询字符串的信息。

3.Cookies：“Cookies”选项卡显示有关请求设置的cookie的信息。

4.标头：“标头”选项卡提供有关标头的信息，该信息由客户端在发出请求时发送。

5.路由：“路由”选项卡提供有关方法的“路由模式”和“路由HTTP动词”类型等信息。

现在，如果您验证“查询”选项卡和“Cookies”选项卡，那么您将看不到任何信息，因为您没有在URL中传递任何查询字符串值，或者未在请求中设置Cookie。

注意：仅当应用程序在开发环境中运行时，才应启用“开发人员异常页面中间件”。当应用程序在生产环境中运行时，您不想共享详细的异常信息。

如何自定义UseDeveloperExceptionPage中间件?

如果需要，还可以自定义UseDeveloperExceptionPage中间件。您需要记住的一点是，每当您要在ASP.NET Core中自定义中间件组件时，都需要使用相应的Options对象。例如
1.UseDeveloperExceptionPage =>使用DeveloperExceptionPageOptions对象自定义此中间件
2.UseDefaultFiles =>使用DefaultFilesOptions对象来自定义此中间件
3.UseStaticFiles =>使用StaticFileOptions对象来自定义此中间件
4.UseFileServer =>使用FileServerOptions对象来自定义此中间件
由于我们将自定义UseDeveloperExceptionPage（）中间件组件，因此我们需要使用DeveloperExceptionPageOptions对象。因此，如下所示修改Startup类的Configure方法。

如您在上面的代码中看到的，我们正在使用一个名为SourceCodeLineCount的属性。DeveloperExceptionPageOptions类的SourceCodeLineCount属性指定在导致异常的代码行之前和之后要包含的代码行数。
现在，如果在进行上述更改的情况下运行应用程序，则将出现以下错误。请查看错误的行号，即37行。同时，请查看错误行之前和之后的行号。

在哪配置UseDeveloperExceptionPage中间件?

我们需要在应用程序的请求处理管道中尽早配置UseDeveloperExceptionPage（）中间件，以便它可以处理未处理的异常，然后显示带有异常详细信息的Developer Exception Page。
让我们看看在导致异常的中间件之后配置UseDeveloperExceptionPage（）中间件时发生了什么。请如下所示修改Configure（）方法。

完成上述更改后，当我们运行该应用程序时，它不会显示开发人员的异常页面，而只会返回默认的错误状态代码。这就是为什么我们需要尽早配置UseDeveloperExceptionPage（）中间件来处理请求处理管道中应用程序未处理的异常的原因。