Diagnosing Runtime Application Issues

Gill Cleeren
ARCHITECT

@gillcleeren   www.snowball.be

Overview

Diagnostics middleware

Logging middleware

Combining with Filters

Adding Azure Application Insights

Diagnostics Middleware

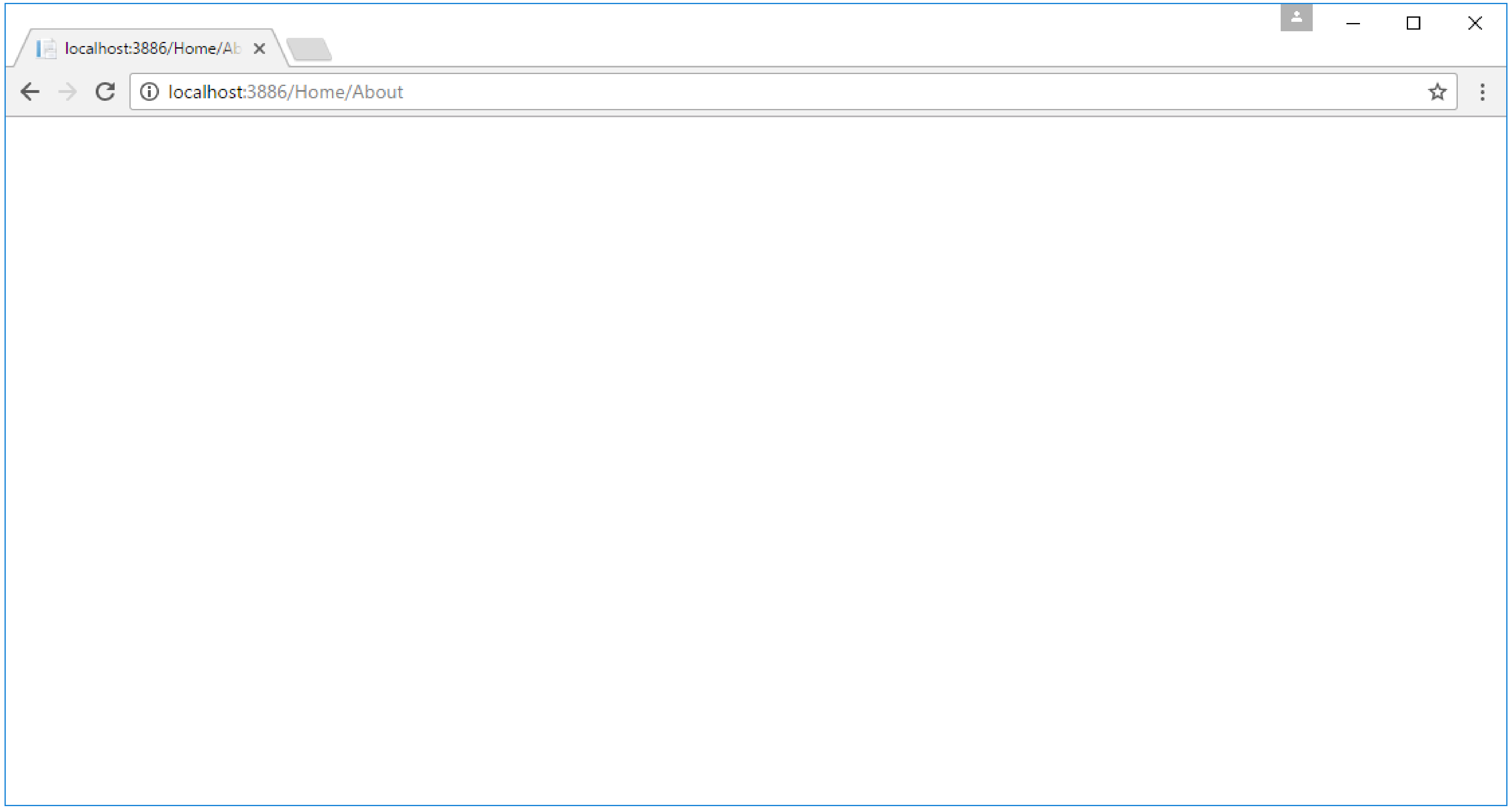
Reporting information and handling exceptions

Microsoft.AspNetCore.Diagnostics

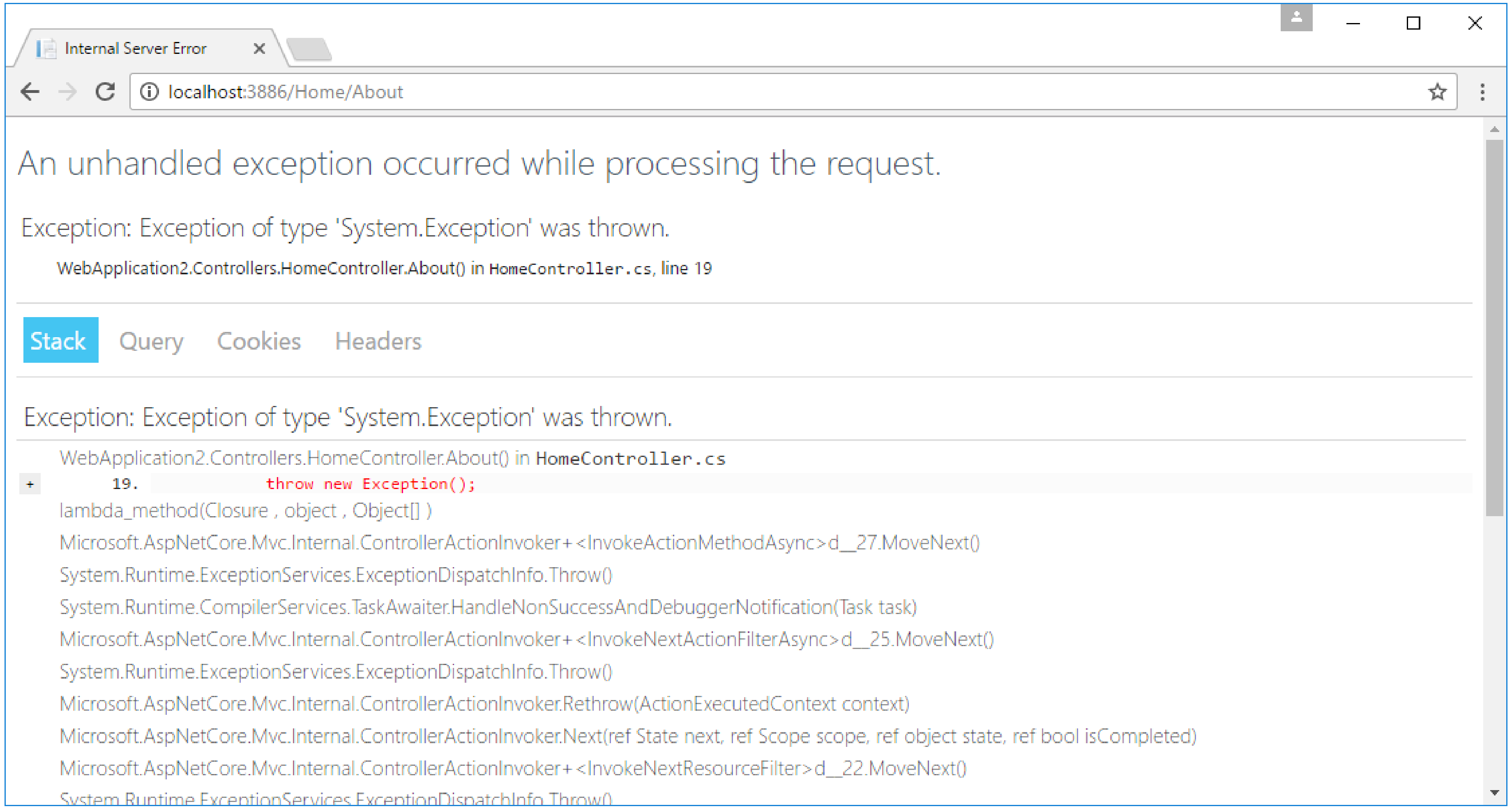
Diagnostics Package

tionPage

Without Diagnostics Enabled



UseDeveloperExceptionPage



An unhandled exception occurred while processing the request.

Exception: Exception of type 'System.Exception' was thrown.

WebApplication2.Controllers.HomeController.About() in `HomeController.cs`, line 19

Exception: Exception of type 'System.Exception' was thrown.

```
WebApplication2.Controllers.HomeController.About() in HomeController.cs
+     19.              throw new Exception();
lambda_method(Closure , object , Object[] )
Microsoft.AspNetCore.Mvc.Internal.ControllerActionInvoker+<InvokeActionMethodAsync>d__27.MoveNext()
System.Runtime.ExceptionServices.ExceptionDispatchInfo.Throw()
System.Runtime.CompilerServices.TaskAwaiter.HandleNonSuccessAndDebuggerNotification(Task task)
Microsoft.AspNetCore.Mvc.Internal.ControllerActionInvoker+<InvokeNextActionFilterAsync>d__25.MoveNext()
System.Runtime.ExceptionServices.ExceptionDispatchInfo.Throw()
Microsoft.AspNetCore.Mvc.Internal.ControllerActionInvoker.Rethrow(ActionExecutedContext context)
Microsoft.AspNetCore.Mvc.Internal.ControllerActionInvoker.Next(ref State next, ref Scope scope, ref object state, ref bool isCompleted)
Microsoft.AspNetCore.Mvc.Internal.ControllerActionInvoker+<InvokeNextResourceFilter>d__22.MoveNext()
System.Runtime.ExceptionServices.ExceptionDispatchInfo.Throw()
```

Diagnostics Package

ti

St

ndler

UseWelcomePage

Demo

Using the different diagnostics options in our site

A question from Bethany

- Some users are complaining they're getting errors on my site. Can we get more information about the errors which are occurring?

Logging Middleware

Built-in Logging Systems in ASP.NET Core

- EventSource
- ILogger
- DiagnosticSource

ILogger is most common in ASP.NET Core

Based on providers

- Built-in
- Extensible using other providers

Built-in Providers

Console

Debug

EventSource

EventLog

TraceSource

Azure App Service

```
public void Configure(ILoggerFactory loggerFactory)
{

    loggerFactory.AddConsole(LogLevel.Debug);
    loggerFactory.AddDebug(LogLevel.Debug);

}
```

Adding a Logging Provider

Logging

```
public class HomeController : Controller

{

    private readonly ILogger<HomeController> _logger;

    public HomeController(ILogger<HomeController> logger)
    {
        _logger = logger;
    }

    ...

    _logger.LogInformation
        (LogEventIds.LoadHomepage, "Loading home page");

}
```

Log Levels

Trace	Debug	Information
Warning	Error	Critical

Other options with ILogger

- Log Category
- Log event ID
- Format string

Demo

Setting up logging in our application

Enabling logging in the controllers

Commonly User Third-party Providers

elmah

NLog

Serilog

```
Log.Logger = new LoggerConfiguration()
    .MinimumLevel.Debug()
    .WriteTo.RollingFile(
       Path.Combine(env.ContentRootPath, "BethanysLogs-
       {Date}.txt"))
    .CreateLogger();

loggerfactory.AddSerilog();
```

Adding Serilog

Rolling log file configuration

Demo

Adding support for using Serilog

Combining with Filters

Allow us to add logic in MVCs request

- Before or after

Used often for cross-cutting concerns

- Avoiding code duplication

Common usages

- Authorization
- Requiring HTTPS

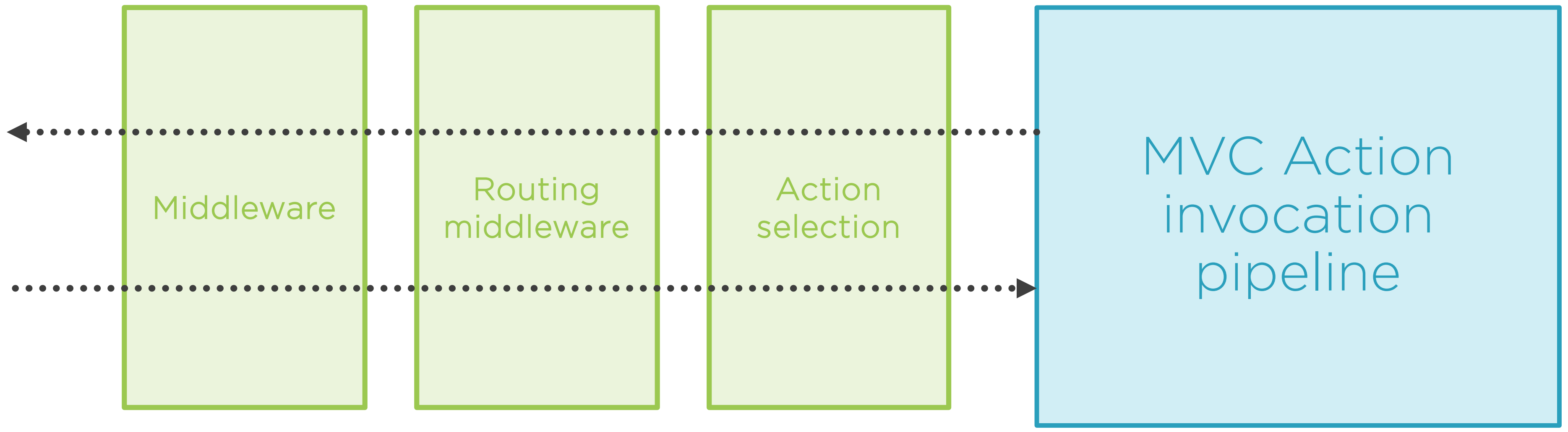
Without Filters

```
public IActionResult Checkout(Order order)
{

    if (!Request.IsHttps)
    {
        return new
            StatusCodeResult(StatusCodes.Status403Forbidden);

    }

    else
    {
        //continue placing the order
    }

}
```

```
[RequireHttps]
public IActionResult Checkout(Order order)
{
    //Continue placing order here
}
```

With a Filter

Filter Execution



Types of Filters

Authorization

Resource

Action

Exception

Result

```
namespace Microsoft.AspNetCore.Mvc.Filters
{
    public interface IFilterMetadata { }
}
```

IFilterMetadata Interface

Authorization Filters

- First to be executed
- Only have before method
- [Authorize]

```
public interface IAuthorizationFilter : IFilterMetadata
{
    void OnAuthorization(AuthorizationFilterContext context);
}
```

IAuthorization Filter

RequireHeaderFilterAttribute

```
public class RequireHeaderFilterAttribute : Attribute, IAuthorizationFilter
{
    public void OnAuthorization(AuthorizationFilterContext context)
    {
        if (!context.HttpContext.Request.Headers.Keys.Contains("Referrer")
            ||context.HttpContext.Request.Headers["Referrer"]
                .Equals("http://www.bethanyspieshop.com"))
        {

            context.Result = new
                StatusCodeResult(StatusCodes.Status403Forbidden);

        }

    }

}
```

Action Filters

- General purpose
- Can interrupt the flow before the action
- Can change the result of an action
- Typically implement the IActionFilter interface

```
public interface IActionFilter : IFilterMetadata
{
    void OnActionExecuting(ActionExecutingContext context);
    void OnActionExecuted(ActionExecutedContext context);
}
```

IActionFilter

Context gives access to

- Controller
- ActionArguments
- Result

Demo

Creating an Authorization Filter

Creating an Action Filter

Creating an Exception Filter

Global filters are applied on all action methods

Any filter can be used for this

Registering a Global Filter

```
services.AddMvc
(
    config =>
    {
        config.Filters.
            AddService(typeof(SomeActionFilterAttribute));
    }

)
```

Demo

Working with Global Filters

Adding Azure Application Insights

Azure Application Insights

- Cloud-based insight in applications
- Information about performance and exceptions
- Alerts and dashboards

Demo

Sending logs and exceptions to Azure Application Insights

Summary

ILogger used for logging information

Filters can be used for cross-cutting concerns

Azure Application Insights can extend the logging feature of the application

Up next:

Improving the site's performance