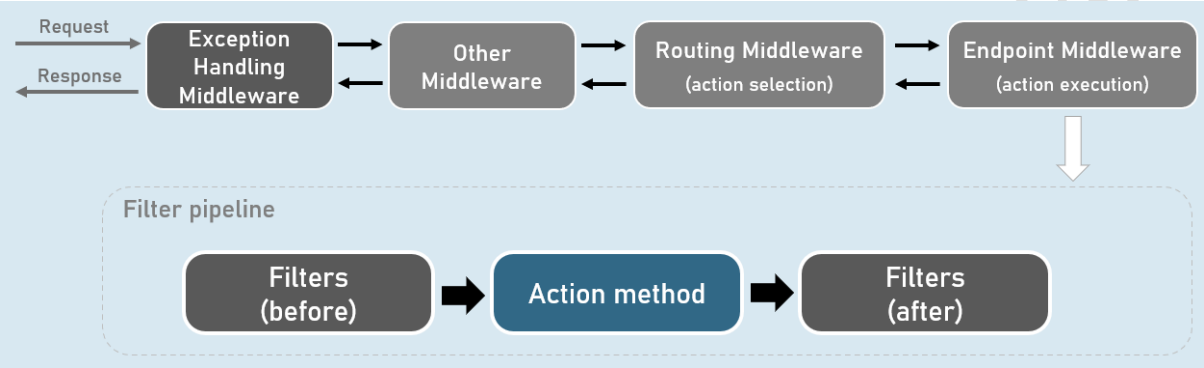
Asp.Net Core – True Ultimate Guide

Section 22 – Error Handling – Cheat Sheet

Exception Handling Middleware

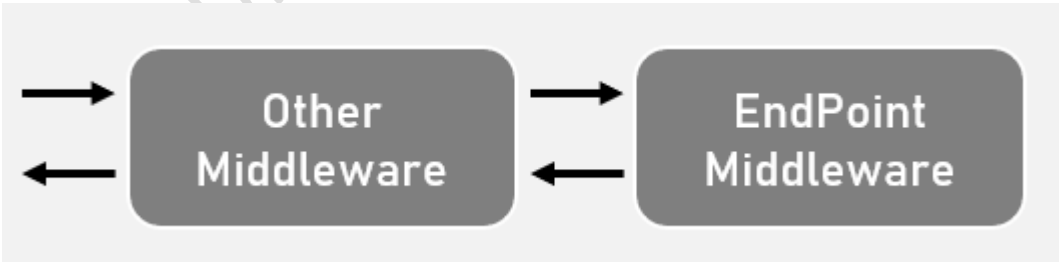
Handles all errors occurred in filter pipeline (including model binding, controllers and filters).

Should be added to the application pipeline, before RoutingMiddleware.



Custom Exception Handling Middleware

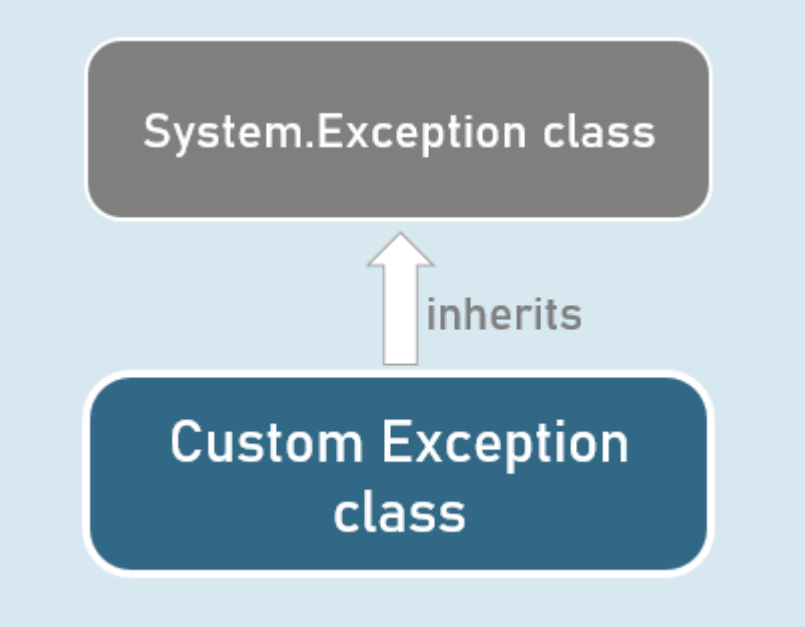
```
public class ExceptionHandlingMiddleware
{
  private readonly RequestDelegate _next; //Stores reference of subsequent middleware


  public ExceptionHandlingMiddleware(RequestDelegate next)
  {
    _next = next;
  }
  public async Task InvokeAsync(HttpContext context)
  {
    try
    {
      await _next(context);
    }
    catch(Exception ex)
    {
      …
    }
  }
}
```



Custom Exceptions

A custom exception class is an exception class that inherits from System.Exception class & represents a domain-specific exception

Used to represent the domain-specific errors stand-out of system-related (.NET) related exceptions.



Custom Exception class

```
public class CustomException : Exception
{
  public CustomException() : base()
  {
  }

  public CustomException(string? message) : base(message)
  {
  }

  public CustomException(string? message, Exception? innerException) : base(message, innerException)
  {
  }
}
```

UseExceptionHandler()

The built-in UseExceptionHandler() middleware redirects to the specified route path, when an unhandled exception occurs during the application execution.

Can be used as an alternative to custom exception handling middleware.

Catches and logs unhandled exceptions.

Re-executes the request in an alternative pipeline using the specified route path.