ASP.NET Core Middleware

Laboratório Web

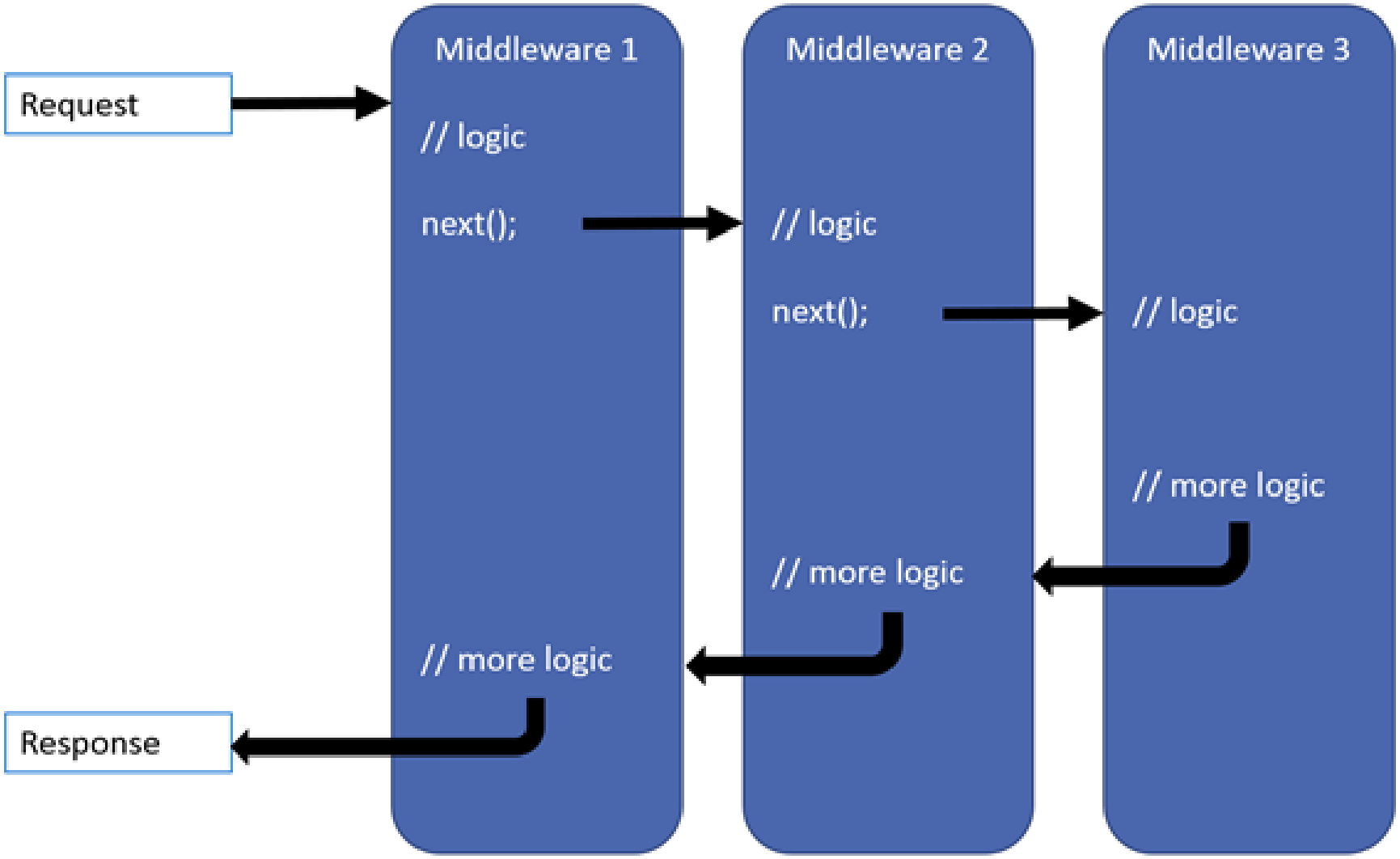
UNIVERSIDADE da MADEIRA

Região Autónoma da Madeira
Governo Regional

startup madeira

Middleware

- É código/software que é executado num pipeline entre o pedido e a resposta
- Pode manipular o pedido e a resposta
- Pode decidir se “passa” o pedido para o próximo componente no pipeline
- Pode ser executado antes ou depois do próximo componente no pipeline


Request
Middleware 1
// logic
next();
// more logic
Middleware 2
// logic
next();
// more logic
Middleware 3
// logic
// more logic
Response

Middleware

- Como utilizar middlewares?

- Já estamos a utilizar ☺

```
// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}
```

```
// Configure the HTTP request pipeline.

app.UseHttpsRedirection();

/// <s
/// Li
/// </
app.Ma
{
```

(extension) IApplicationBuilder IApplicationBuilder.UseHttpsRedirection()
Adds middleware for redirecting HTTP Requests to HTTPS.

Returns:
The IApplicationBuilder for HttpsRedirection.

Middleware

- Como utilizar middlewares?

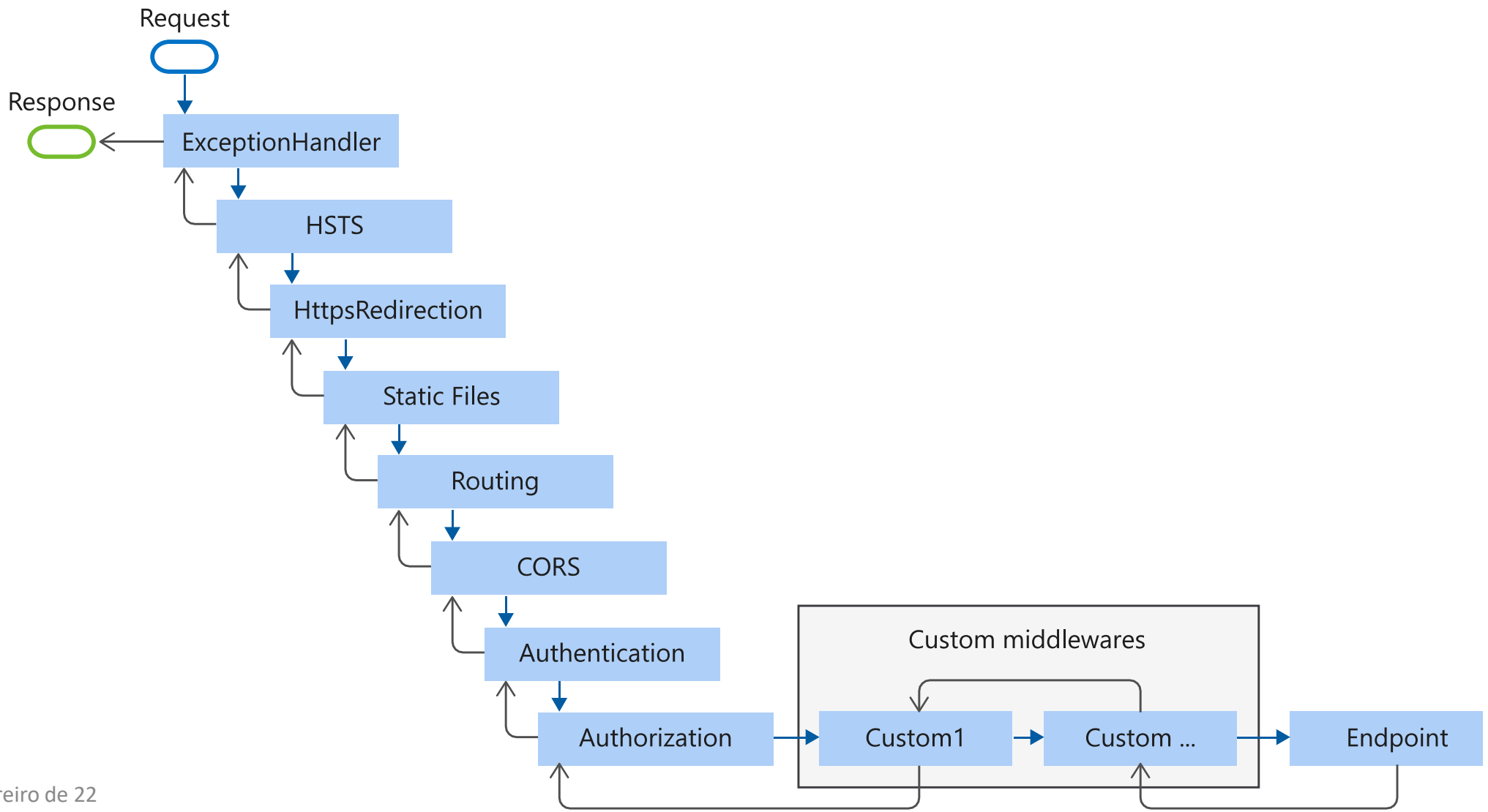
- Já estamos a utilizar ☺

```
app.UseHttpsRedirection();
app.UseStaticFiles();
// app.UseCookiePolicy();

app.UseRouting();
app.UseRequestLocalization();
app.UseCors();

app.UseAuthentication();
app.UseAuthorization();
app.UseSession();
app.UseResponseCompression();
app.UseResponseCaching();
```

Pipeline para um pedido em ASP.NET Core



Built-in middleware

Authentication	Provides authentication support.
Authorization	Provides authorization support.
Cookie Policy	Tracks consent from users for storing personal information and enforces minimum standards for cookie fields, such as secure and SameSite.
CORS	Configures Cross-Origin Resource Sharing.
DeveloperExceptionPage	Generates a page with error information that is intended for use only in the Development environment.
Diagnostics	Several separate middlewares that provide a developer exception page, exception handling, status code pages, and the default web page for new apps.
Forwarded Headers	Forwards proxied headers onto the current request.
Health Check	Checks the health of an ASP.NET Core app and its dependencies, such as checking database availability.
Header Propagation	Propagates HTTP headers from the incoming request to the outgoing HTTP Client requests.
HTTP Logging	Logs HTTP Requests and Responses.

Built-in middleware

HTTP Method Override	Allows an incoming POST request to override the method.
HTTPS Redirection	Redirect all HTTP requests to HTTPS.
HTTP Strict Transport Security (HSTS)	Security enhancement middleware that adds a special response header.
MVC	Processes requests with MVC/Razor Pages.
OWIN	Interop with OWIN-based apps, servers, and middleware.
Response Caching	Provides support for caching responses.
Response Compression	Provides support for compressing responses.
Request Localization	Provides localization support.
Endpoint Routing	Defines and constrains request routes.
SPA	Handles all requests from this point in the middleware chain by returning the default page for the Single Page Application (SPA)
Session	Provides support for managing user sessions.
Static Files	Provides support for serving static files and directory browsing.
URL Rewrite	Provides support for rewriting URLs and redirecting requests.
W3CLogging	Generates server access logs in the W3C Extended Log File Format.
WebSockets	Enables the WebSockets protocol.

Custom middleware - INLINE

```
app.Use(async (context, next) =>
{
    Debug.WriteLine("BEFORE FIRST MIDDLEWARE");
    await next.Invoke();
    Debug.WriteLine("AFTER FIRST MIDDLEWARE");
});


app.Use(async (context, next) =>
{
    Debug.WriteLine("BEFORE SECOND MIDDLEWARE");
    await next.Invoke();
    Debug.WriteLine("AFTER SECOND MIDDLEWARE");
});
```

- A ordem em que adicionamos o middleware interessa
- **context** é o HttpContext, tem o pedido e a resposta http
- **next** é um delegate
- delegate é uma referência para uma função
- **next** – representa o próximo middleware a ser invocado

Custom middleware - CLASS

```
public class CustomMiddleware
{
    private readonly RequestDelegate next;
    public CustomMiddleware(RequestDelegate next)
    {
        this.next = next;
    }
    public async Task InvokeAsync(HttpContext context)
    {
        // Call the next delegate/middleware in the pipeline.
        await next(context);
    }
}
```

- construtor com parâmetro do tipo **RequestDelegate**
- método público com nome **Invoke** ou **InvokeAsync**
  - tem que devolver uma **Task**
  - o primeiro parâmetro tem que ser do tipo **HttpContext**
  - parâmetros extra são adicionados através de **Dependency Injection**

Como utilizar o custom middleware?

```
public static class CustomMiddlewareExtensions
{
  public static IApplicationBuilder UseCustomMiddleware(
    this IApplicationBuilder builder)
    {
      return builder.UseMiddleware<CustomMiddleware>();
    }
}

// Program.cs
app.UseCustomMiddleware();
```

- Adicionámos um método de extensão para expôr o middleware criado atráves da interface **IApplicationBuilder**
- Para introduzir o middleware ao pipeline invocámos o método de extensão a partir da instância da nossa aplicação

Referências

https://docs.microsoft.com/en-us/aspnet/core/fundamentals/middleware/?view=aspnetcore-6.0#middleware-order

We use cookies to improve your experience on our websites and for advertising. Privacy Statement

Microsoft | Docs Documentation Learn Q&A Code Samples Shows Events

ASP.NET Languages Workloads APIs Resources

Docs / .NET / ASP.NET Core / Fundamentals / Middleware

Version

ASP.NET Core 6.0

Filter by title

- ASP.NET Core documentation
- What's new in ASP.NET Core docs
- Overview
- Get started
- What's new
- Tutorials
- Fundamentals
  - Overview
  - App start up
  - Dependency injection (services)
  - Middleware

ASP.NET Core Middleware

Article • 12/14/2021 • 26 minutes to read • +7

Is this page helpful?

By Rick Anderson and Steve Smith

Middleware is software that's assembled into an app pipeline to handle requests and responses. Each component:

- Chooses whether to pass the request to the next component in the pipeline.
- Can perform work before and after the next component in the pipeline.

Request delegates are used to build the request pipeline. The request delegates handle each HTTP request.

Request delegates are configured using Run, Map, and Use extension methods. An individual request delegate can be specified in-line as an anonymous method (called in-line middleware), or it can be defined in a reusable class. These reusable classes and in-line anonymous methods are *middleware*, also called *middleware components*. Each middleware component in the request pipeline is responsible for invoking the next component in the pipeline or short-circuiting the pipeline. When a middleware short-circuits, it's called a *terminal middleware* because it prevents further middleware from processing the request.


qualificar
TAL
A <PROGRAMAÇÃ