## 基于工厂的中间件需要继承IMiddleware接口，在请求到来时，会从IMiddlewareFactory中获取中间件实例

如果中间件没有继承IMiddleware接口，则不会从IMiddlewareFactory中获取

.netcore具有IMiddlewareFactory的默认实现

## **IMiddleware**

使用基于工厂的中间件（继承IMiddleware接口）：

public class FactoryActivatedMiddleware : IMiddleware

{

    private readonly AppDbContext \_db;

    public FactoryActivatedMiddleware(AppDbContext db)

    {

        \_db = db;

}

    public async Task InvokeAsync(HttpContext context, RequestDelegate next)

    {

        var keyValue = context.Request.Query["key"];

        if (!string.IsNullOrWhiteSpace(keyValue))

        {

            \_db.Add(new Request()

                {

                    DT = DateTime.UtcNow,

                    MiddlewareActivation = "FactoryActivatedMiddleware",

                    Value = keyValue

                });

            await \_db.SaveChangesAsync();

        }

        await next(context);

    }

}

**为中间件创建扩展**

public static class MiddlewareExtensions

{

    public static IApplicationBuilder UseFactoryActivatedMiddleware(

        this IApplicationBuilder builder)

    {

        return builder.UseMiddleware<FactoryActivatedMiddleware>();

    }

}

**将工厂中间件添加内置容器中**

public void ConfigureServices(IServiceCollection services)

{

    ...

    services.AddTransient<FactoryActivatedMiddleware>();

}

**应用中间件**

public void Configure (IApplicationBuilder app, IHostingEnvironment env) {

    ...

    app.UseConventionalMiddleware ();

    ...

## **IMiddlewareFactory**

[IMiddlewareFactory](https://docs.microsoft.com/zh-cn/dotnet/api/microsoft.aspnetcore.http.imiddlewarefactory) 作为域服务注册在容器中