包：Microsoft.AspNetCore.StaticFiles

静态文件中间件允许以Http Get的形式访问我们服务器的资源文件

**使用静态文件中间件**

new WebHostBuilder ()

    .UseContentRoot (Directory.GetCurrentDirectory ())

    .UseKestrel ()

    // 使用静态文件中间件

    .Configure (app => app.UseStaticFiles ())

    .Build ()

    .Run ();

当我们不传参数时，默认使用HostingEnvironment的WebRootFileProvider属性作为文件提供者（即，以HostingEnvironment的WebRootPath作为静态文件的根目录）

**使用多个静态文件根目录**

public static void Main ()

{

    string contentRoot = Directory.GetCurrentDirectory ();

    new WebHostBuilder ()

        .UseContentRoot (contentRoot)

        .UseKestrel ()

        .Configure (app => app

            // 使用 WebRoot 为静态文件根目录

            .UseStaticFiles ()

            // 使用 doc 目录为静态文件目录，映射的 url前缀 为 /documents

            .UseStaticFiles (new StaticFileOptions {

                FileProvider = new PhysicalFileProvider (Path.Combine (contentRoot, "doc")),

                RequestPath = "/documents"

            }))

        .Build ()

        .Run ();

}

**使用目录浏览**

当url为 /documents/ 的形式时，其是想访问我们的 doc 目录，如下中间件启动目录浏览

public class Program {

    public static void Main () {

        string contentRoot = Directory.GetCurrentDirectory ();

        IFileProvider fileProvider = new PhysicalFileProvider (

            Path.Combine (contentRoot, "doc"));

        new WebHostBuilder ()

            .UseContentRoot (contentRoot)

            .UseKestrel ()

            .Configure (app => app

                // 使用静态文件中间件

                .UseStaticFiles ()

                .UseStaticFiles (new StaticFileOptions {

                    FileProvider = fileProvider,

                    RequestPath = "/documents"

                })

                // 使用目录浏览中间件

                .UseDirectoryBrowser ()

                .UseDirectoryBrowser (new DirectoryBrowserOptions {

                    FileProvider = fileProvider,

                    RequestPath = "/documents"

                }))

            .Build ()

            .Run ();

    }

}

**默认显示页**

当url为 /documents/ 的形式时，我们期望返回的是 /doc 目录下的 index.html 时，应该使用默认页中间件

public class Program {

    public static void Main () {

        string contentRoot = Directory.GetCurrentDirectory ();

        IFileProvider fileProvider = new PhysicalFileProvider (Path.Combine (contentRoot, "doc"));

        new WebHostBuilder ()

            .UseContentRoot (contentRoot)

            .UseKestrel ()

            .Configure (app => app

                // 使用默认页

                .UseDefaultFiles ()

                .UseDefaultFiles (new DefaultFilesOptions {

                    RequestPath = "/documents",

                    FileProvider = fileProvider,

                })

                // 使用静态文件

                .UseStaticFiles ()

                .UseStaticFiles (new StaticFileOptions {

                    FileProvider = fileProvider,

                        RequestPath = "/documents"

                })

                // 使用目录浏览

                .UseDirectoryBrowser ()

                .UseDirectoryBrowser (new DirectoryBrowserOptions {

                    FileProvider = fileProvider,

                    RequestPath = "/documents"

                }))

            .Build ()

            .Run ();

    }

}

如上可见，只有不存在默认页时，才会触发目录浏览

**媒体类型映射**

响应报文必须设置Content-Type报头，静态文件中间件是如何获取文件的Content-Type（内容类型）的？

静态文件通过IContentTypeProvider获取文件扩展名对应的Content-Type

FileExtensionContentTypeProvider是IContentTypeProvider的默认实现

我们通过如下代码，实现新的扩展名映射到Content-Type

public class Program {

    public static void Main () {

        FileExtensionContentTypeProvider contentTypeProvider = new FileExtensionContentTypeProvider ();

        // 添加新的映射

        contentTypeProvider.Mappings.Add (".img", "image/jpg");

        new WebHostBuilder ()

            .UseContentRoot (Directory.GetCurrentDirectory ())

            .UseKestrel ()

            .Configure (app => app.UseStaticFiles (new StaticFileOptions {

                // 指定内容类型提供者

                ContentTypeProvider = contentTypeProvider

            }))

            .Build ()

            .Run ();

    }

}