ASP.NET Core 基础系列（5）(appSetting)

原创 痕迹 CodeShare 6天前

收录于话题
#ASP.NET Core

8个

appSetting.json

当我们使用空项目模板或Razor页面或MVC模板或Web API模板创建ASP.NET Core Web应用程序时，Visual Studio会自动为我们创建appsettings.json文件，如下图所示。

appsettings.json文件是一个应用程序配置文件，用于存储配置设置，例如数据库连接字符串，任何应用程序范围的全局变量等。如果打开ASP.NET Core appsettings.json文件，则默认情况下会看到以下代码 这是由Visual Studio创建的。

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*"
}
```

现在，在该文件中添加一个名为MyName的键。
为此，请如下所示修改appsettings.json文件。
注: 由于它是JSON文件，因此您需要以键值对的形式存储值。

访问appSetting.json

若要访问Startup类中的配置信息，需要使用ASP.NET Core框架提供的IConfiguration服务。
因此，只需要做的只是通过Startup类的构造函数注入IConfiguration服务。
为此，请修改Startup.cs文件中存在的Startup类，如下所示。

```
public class Startup
{
    private IConfiguration config;

    0 个引用
    public Startup(IConfiguration config)
    {
        this.config = config;
    }

    0 个引用
    public void ConfigureServices(IServiceCollection services)...

    0 个引用
    public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
    {
        if (env.IsDevelopment()) //是否处于开发模式
        {
            //异常中间件
            app.UseDeveloperExceptionPage();
        }
        //路由
        app.UseRouting();
        //终结点
        app.UseEndpoints(endpoints =>
        {
            endpoints.MapGet("/", async context =>
            {
                //输出当前进程名称
                await context.Response.WriteAsync(config["MyName"]);
            });
        });
    }
}
```

首先，我们创建了一个IConfiguration _config类型的私有变量（此IConfiguration接口属于Microsoft.Extensions.Configuration命名空间，因此请首先使用此命名空间）。

然后，通过构造函数依赖项注入，我们注入IConfiguration对象并将其存储在私有变量config中, 以下代码。

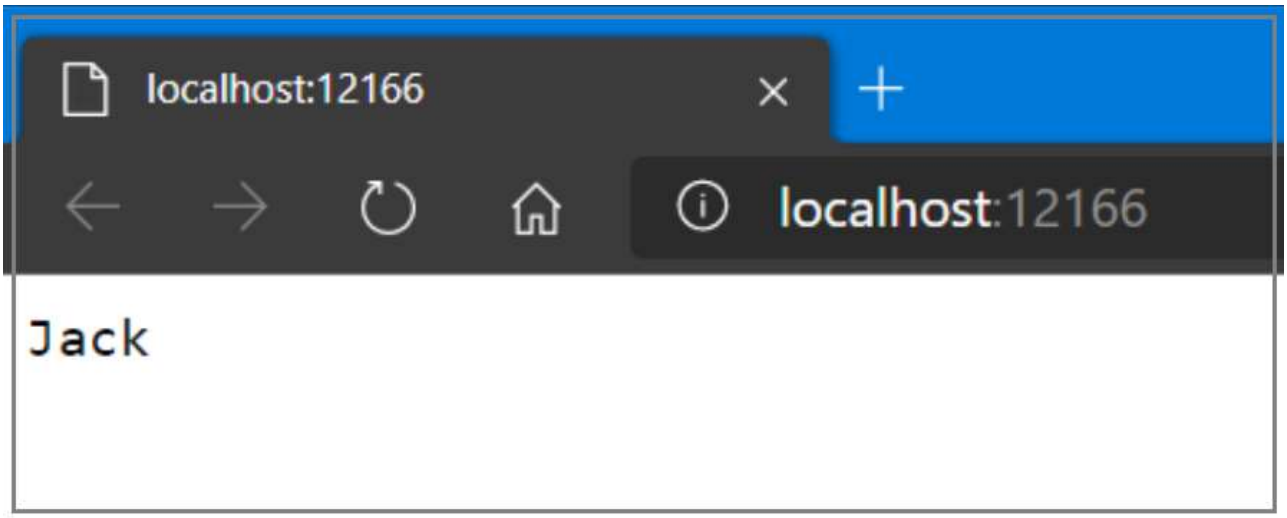
```
private IConfiguration config;

0 个引用
public Startup(IConfiguration config)
{
    this.config = config;
}
```

然后，我们使用IConfiguration服务实例访问配置变量，即MyName, 以下代码。

```
app.UseEndpoints(endpoints =>
{
    endpoints.MapGet("/", async context =>
    {
        await context.Response.WriteAsync(config["MyName"]);
    });
});
```

现在运行该应用程序，您应该在浏览器中看到预期的值，如下图所示。



访问配置的默认顺序是什么?

为同一键读取各种配置源的默认顺序如下:

1. appsettings.json
2. appsettings.{Environment}.json
3. 用户设置
4. 环境变量
5. 命令行参数
   如您所见，Program类的Main（）方法调用CreateHostBuilder（）方法。然后，CreateHostBuilder（）方法在Host类上调用CreateDefaultBuilder（）方

法。
此CreateDefaultBuilder（）方法是设置读取所有配置源的默认顺序的方法。
如果需要，还可以更改此默认顺序，或者即使您愿意，也可以添加自己的自定义配置源以及现有的配置源。