**启动示例代码**

public static void Main (string[] args) {

    // WebHost生成器

    IWebHostBuilder webHostBuilder = new WebHostBuilder ();

    // 注册 Kestrel 服务器

    webHostBuilder.UseKestrel ();

    // 使用 Url

    webHostBuilder.UseUrls ("https://localhost:5000");

    // 注册中间件

    webHostBuilder.UseStartup<Startup> ();

    // 生成WebHost

    IWebHost webHost = webHostBuilder.Build ();

    // 运行WebHost

    webHost.Run ();

}

**IWebHostBuilder**

public interface IWebHostBuilder {

    // 生成 IWebHost

    IWebHost Build ();

    // 向依赖注入容器注册服务

    IWebHostBuilder ConfigureServices (Action<IServiceCollection> configureServices);

    // 获取配置

    string GetSetting (string key);

    // 设置配置

    IWebHostBuilder UseSetting (string key, string value);

}

**UseKestrel () 做了什么**

UseKestrel调用IWebHostBuilder的ConfigureServices方法，向依赖注入容器注册一个IServer服务器

hostBuilder.ConfigureServices (services => {

    ...

    services.AddSingleton<IServer, KestrelServer> ();

});

**UseUrls ()做了什么**

UseUrls调用IWebHostBuilder的UseSettion方法将我们传入的urls保存到一个键值对中

hostBuilder.UseSetting (

    WebHostDefaults.ServerUrlsKey,

    string.Join (ServerUrlsSeparator, urls));

**UseStartup()做了什么**

UseStartup调用IWebHostBuilder的ConfigureServices方法，向依赖注入容器注册我们指定的Startup类型

hostBuilder.ConfigureServices (services => {

    if (typeof (IStartup).GetTypeInfo ().IsAssignableFrom (startupType.GetTypeInfo ())) {

        services.AddSingleton (typeof (IStartup), startupType);

    }

    ...

});

**Build () 做了什么**

Build () 新建一个 WebHost，并创建IServiceCollection

调用ConfigureServices注册的Action向IServiceCollection添加服务

通过IServiceCollection获取IStartup的实例，然后调用IStartup的ConfigureServices，然后利用ConfigureServices返回的IServiceProvider作为站点的依赖注入容器

（注：如果我们编写的Startup的ConfigureServices返回是void，Build会利用IServiceCollection生成IServiceProvider）

**Run() 做了什么**

Run() 调用WebHost的Start方法

WebHost的Start方法做了什么：

新建了一个IApplicationBuilder传给你IStartup实例的Configure方法注册中间件

调用IApplicationBuilder的build生成RequestDelegate（封装了所有中间件的处理流程）

新建一个IHttpApplication（代表一个应用）并将RequestDelegate传给它

从依赖注入容器获取IServer，将IHttpApplication作为参数调用其StartAsync方法启动服务