**涉及概念**

本文主要介绍IConfiguration、ConfigurationBuilder、IConfigurationSource的之间的关系和用法，如果你已经了解其用途，可以跳过

IConfiguration：配置，其包含整个程序的配置数据

IConfigurationSource：配置源，其有多种实现，包括Json、ini、xml等

ConfigurationBuilder：配置生成器，其利用IConfigurationSource生成IConfiguration

**基本示例**

// IConfiguration使用 : 来分隔层次结构，如下配置结构为

// {

// format:{

// dateTime:{

// longDatePattern: "dddd, MMMM d, yyyy",

// longTimePattern: "h:mm:ss tt"

// ...

// }

// }

// }

Dictionary<string, string> source = new Dictionary<string, string>

{

["format:dateTime:longDatePattern"] = "dddd, MMMM d, yyyy",

["format:dateTime:longTimePattern"] = "h:mm:ss tt",

["format:dateTime:shortDatePattern"] = "M/d/yyyy",

["format:dateTime:shortTimePattern"] = "h:mm tt",

};

// 配置

IConfiguration config =

// 新建ConfigurationBuilder

new ConfigurationBuilder()

// 添加内存配置源

.Add(new MemoryConfigurationSource { InitialData = source })

// 添加Json配置源

.AddJsonFile("./appsettings.json")

// 生成Configuration

.Build();

// 使用 GetSection 获取数据

var longDatePattern = config.GetSection("format").GetSection("dateTime").GetValue<string>("longDatePattern");

Console.WriteLine(longDatePattern);

// 当然，你也可以直接获取数据， : 代表层次结构

var longDatePattern2 = config["format:dateTime:longDatePattern"];

Console.WriteLine(longDatePattern2);

**配置源**

.net core 提供的配置源有下面几种

1. 内存配置源

内存配置源的实现为MemoryConfigurationSource，其提供了2个扩展方法将源快速添加到ConfigurationBuilder中

public static IConfigurationBuilder AddInMemoryCollection(

this IConfigurationBuilder configurationBuilder);

public static IConfigurationBuilder AddInMemoryCollection(

this IConfigurationBuilder configurationBuilder,

IEnumerable<KeyValuePair<string, string>> initialData);

1. 环境变量配置源

环境变量配置源的实现为EnvironmentVariablesConfigurationSource

包：Microsoft.Extensions.Configuration.EnvironmentVariables

public static IConfigurationBuilder AddEnvironmentVariables(

this IConfigurationBuilder configurationBuilder);

public static IConfigurationBuilder AddEnvironmentVariables(

this IConfigurationBuilder configurationBuilder,

string prefix);

1. 命令行参数

命令行参数配置源的实现为CommandLineConfigurationSource

包：Microsoft.Extensions.Configuration.CommandLine

public static IConfigurationBuilder AddCommandLine(

this IConfigurationBuilder configurationBuilder,

string[] args);

public static IConfigurationBuilder AddCommandLine(

this IConfigurationBuilder configurationBuilder,

string[] args,

IDictionary<string, string> switchMappings);

1. Json文件

其实现的配置源为JsonConfigurationSource

包：Microsoft.Extensions.Configuration.Json

包提供的添加Json配置源的扩展

public static class JsonConfigurationExtensions

{

public static IConfigurationBuilder AddJsonFile(

this IConfigurationBuilder builder,

Action<JsonConfigurationSource> configureSource);

public static IConfigurationBuilder AddJsonFile(

this IConfigurationBuilder builder,

string path);

public static IConfigurationBuilder AddJsonFile(

this IConfigurationBuilder builder,

string path,

bool optional);

public static IConfigurationBuilder AddJsonFile(

this IConfigurationBuilder builder,

string path, // 配置源路径

bool optional, // 是否可选配置源

bool reloadOnChange); // 改变是否重新载入

}

1. Xml文件

其实现为XmlConfiguationSource

包：Microsoft.Extensions.Configuration.Xml

包提供的XML配置源的扩展

public static class XmlConfigurationExtensions

{

public static IConfigurationBuilder AddXmlFile(

this IConfigurationBuilder builder,

string path,

bool optional,

bool reloadOnChange);

...

}

XML文件示例

<Profiles>

<Foo Gender="Male" Age="18">

<ContactInfo EmailAddress ="foobar@outlook.com" PhoneNo="123"/>

</Foo>

<Bar Gender="Male" Age="25">

<ContactInfo EmailAddress ="foobar@outlook.com" PhoneNo="123"/>

</Bar>

<Baz Gender="Male" Age="18">

<ContactInfo EmailAddress ="baz@outlook.com" PhoneNo="789"/>

</Baz>

</Profiles>

1. ini文件

其实现为IniConfigurationSource

包：Microsoft.Extensions.Configuration.Ini

包提供的扩展

public static class IniConfigurationExtensions

{

public static IConfigurationBuilder AddIniFile(

this IConfigurationBuilder builder,

string path,

bool optional,

bool reloadOnChange);

...

}

INI文件示例

Gender = "Male"

Age = "18"

ContactInfo:EmailAddress = "foobar@outlook.com"

ContactInfo:PhoneNo = "123456789"

**Asp.net core**

1. 添加配置源

我们可以在IHostBuilder.ConfigureHostConfiguration中添加配置源

Host.CreateDefaultBuilder(args)

    .ConfigureHostConfiguration((config) => {

        Dictionary<string, string> source = new Dictionary<string, string>

        {

            ["format:dateTime:longDatePattern"] = "dddd, MMMM d, yyyy",

            ["format:dateTime:longTimePattern"] = "h:mm:ss tt",

            ["format:dateTime:shortDatePattern"] = "M/d/yyyy",

            ["format:dateTime:shortTimePattern"] = "h:mm tt",

        };

        config.AddInMemoryCollection(source);

    })

1. 使用配置

我们可以在任何地方使用依赖注入获取IConfiguration

public Startup(IConfiguration configuration)

{

    var longDatePattern2 = configuration["format:dateTime:longDatePattern"];

}