# [Asp.net Core中使用Session](http://www.cnblogs.com/sword-successful/p/6243841.html)

### **前言**

      2017年就这么悄无声息的开始了，2017年对我来说又是特别重要的一年。

      元旦放假在家写了个Asp.net Core验证码登录， 做demo的过程中遇到两个小问题，第一是在Asp.net Core中引用dll，以往我们引用DLL都是直接引用，在Core里这样是不行的，必须基于NuGet添加，或者基于project.json添加，然后保存VS会启动还原类库。

第二就是使用Session的问题，Core里使用Session需要添加Session类库。

### **添加Session**

     在你的项目上基于NuGet添加：Microsoft.AspNetCore.Session。

### **修改startup.cs**

    在startup.cs找到方法ConfigureServices(IServiceCollection services) 注入Session(这个地方是Asp.net Core pipeline):services.AddSession();

    接下来我们要告诉Asp.net Core使用内存存储Session数据，在Configure(IApplicationBuilder app,...)中添加代码:app.UserSession();

### **Session**

1、在MVC Controller里使用HttpContext.Session

[IMG_256](http://www.cnblogs.com/sword-successful/p/javascript:void(0);" \o "复制代码)

using Microsoft.AspNetCore.Http;

public class HomeController:Controller

{

public IActionResult Index()

{

HttpContext.Session.SetString("code","123456");

return View();

}

public IActionResult About()

{

ViewBag.Code=HttpContext.Session.GetString("code");

return View();

}

}

[IMG_257](http://www.cnblogs.com/sword-successful/p/javascript:void(0);" \o "复制代码)

2、如果不是在Controller里，你可以注入IHttpContextAccessor

[IMG_258](http://www.cnblogs.com/sword-successful/p/javascript:void(0);" \o "复制代码)

public class SomeOtherClass

{

private readonly IHttpContextAccessor \_httpContextAccessor;

private ISession \_session=> \_httpContextAccessor.HttpContext.Session;

public SomeOtherClass(IHttpContextAccessor httpContextAccessor)

{

\_httpContextAccessor=httpContextAccessor;

}

public void Set()

{

\_session.SetString("code","123456");

}

public void Get()

{

string code = \_session.GetString("code");

}

}

[IMG_259](http://www.cnblogs.com/sword-successful/p/javascript:void(0);" \o "复制代码)

### **存储复杂对象**

      存储对象时把对象序列化成一个json字符串存储。

[IMG_260](http://www.cnblogs.com/sword-successful/p/javascript:void(0);" \o "复制代码)

public static class SessionExtensions

{

public static void SetObjectAsJson(this ISession session, string key, object value)

{

session.SetString(key, JsonConvert.SerializeObject(value));

}

public static T GetObjectFromJson<T>(this ISession session, string key)

{

var value = session.GetString(key);

return value == null ? default(T) : JsonConvert.DeserializeObject<T>(value);

}

}

[IMG_261](http://www.cnblogs.com/sword-successful/p/javascript:void(0);" \o "复制代码)

var myComplexObject = new MyClass();

HttpContext.Session.SetObjectAsJson("Test", myComplexObject);

var myComplexObject = HttpContext.Session.GetObjectFromJson<MyClass>("Test");

### **使用SQL Server或Redis存储**

1、SQL Server

添加引用  "Microsoft.Extensions.Caching.SqlServer": "1.0.0"

注入：

[IMG_262](http://www.cnblogs.com/sword-successful/p/javascript:void(0);" \o "复制代码)

// Microsoft SQL Server implementation of IDistributedCache.// Note that this would require setting up the session state database.

services.AddSqlServerCache(o =>

{

o.ConnectionString = "Server=.;Database=ASPNET5SessionState;Trusted\_Connection=True;";

o.SchemaName = "dbo";

o.TableName = "Sessions";

});

[IMG_263](http://www.cnblogs.com/sword-successful/p/javascript:void(0);" \o "复制代码)

2、Redis

添加引用   "Microsoft.Extensions.Caching.Redis": "1.0.0"

注入：

// Redis implementation of IDistributedCache.// This will override any previously registered IDistributedCache service.

services.AddSingleton<IDistributedCache, RedisCache>();

http://www.cnblogs.com/sword-successful/p/6243841.html