大家在使用ASP.NET的時候一定都用過FormsAuthentication做登入使用者的身份認證，FormsAuthentication的核心就是Cookie，ASP.NET會將使用者名稱儲存在Cookie中。

現在到了ASP.NET CORE的時代，但是ASP.NET CORE中沒有FormsAuthentication這個東西，那麼怎麼做身份認證呢？答案是ASP.NET CORE已經為我們內建了Cookie身份認證的功能，而且使用起來非常方便，注意本文是基於ASP.NET CORE 2.0版本來闡述Cookie認證方式的。

1.從ASP.NET CORE OWIN框架中啟用Cookie身份認證功能

要在ASP.NET CORE中使用Cookie身份認證，第一步就是在專案中的OWIN框架檔案Startup.cs中啟用Cookie身份認證中介軟體。

首先我們在Startup中的ConfigureServices方法中使用services.AddAuthentication註冊Cookie認證服務，如下程式碼所示：

**public** **void** **ConfigureServices**(IServiceCollection services)

{

services.AddMvc();

//註冊Cookie認證服務

services.AddAuthentication(CookieAuthenticationDefaults.AuthenticationScheme).AddCookie();

}

然後在Startup中的Configure方法中使用app.UseAuthentication啟用Cookie認證中介軟體（注意其中app.UseAuthentication和app.UseMvc的呼叫順序不能反），如下程式碼所示：

**public** **void** **Configure**(IApplicationBuilder app, IHostingEnvironment env)

{

**if** (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

app.UseBrowserLink();

}

**else**

{

app.UseExceptionHandler("/Home/Error");

}

app.UseStaticFiles();

//注意app.UseAuthentication方法一定要放在下面的app.UseMvc方法前面，否者後面就算呼叫HttpContext.SignInAsync進行使用者登入後，使用

//HttpContext.User還是會顯示使用者沒有登入，並且HttpContext.User.Claims讀取不到登入使用者的任何資訊。

//這說明Asp.Net OWIN框架中MiddleWare的呼叫順序會對系統功能產生很大的影響，各個MiddleWare的呼叫順序一定不能反

app.UseAuthentication();

app.UseMvc(routes =>

{

routes.MapRoute(

name: "default",

**template**: "{controller=Home}/{action=Index}/{id?}");

});

}

2.登入使用者

在ASP.NET CORE中使用Cookie認證登入使用者的方法和傳統的FormsAuthentication不太一樣，大致步驟如下：

建立Claim型別的陣列，將登入使用者的所有資訊（比如使用者名稱）儲存在Claim型別的字串鍵值對中  
將上面建立的Claim型別的陣列傳入ClaimsIdentity中，用來構造一個ClaimsIdentity物件  
將上面建立的ClaimsIdentity物件傳入ClaimsPrincipal中，用來構造一個ClaimsPrincipal物件  
呼叫HttpContext.SignInAsync方法，傳入上面建立的ClaimsPrincipal物件，完成使用者登入  
所以我們可以看到整個ASP.NET CORE的Cookie認證登入流程比以前ASP.NET的FormsAuthentication還是要複雜許多，畢竟以前一個FormsAuthentication.SetAuthCookie方法就搞定了。

在本文的例子中我們在專案中預設的HomeController中建立了一個Acion方法Login，來實現使用者登入的程式碼。當然這裡我們實現的是最簡的Cookie登入，下面程式碼中實際上還可以設定Cookie是否持久化、Cookie多久過期、儲存登入使用者資訊的Cookie的名字是什麼等，我們就不做過多介紹了，大家可以閱讀本文最後推薦的兩份官方文件瞭解更多。

Login方法的程式碼如下：

**///** **<summary>**

**///** 該Action登入使用者Wangdacui到Asp.Net Core

**///** **</summary>**

**public** IActionResult **Login**() {

//下面的變數claims是Claim型別的陣列，Claim是string型別的鍵值對，所以claims陣列中可以儲存任意個和使用者有關的資訊，

//不過要注意這些資訊都是加密後儲存在客戶端瀏覽器cookie中的，所以最好不要儲存太多特別敏感的資訊，這裡我們只儲存了使用者名稱到claims陣列,

//表示當前登入的使用者是誰

**var** claims = **new**[] {

**new** Claim("UserName", "Wangdacui")

};

**var** claimsIdentity = **new** ClaimsIdentity(claims, CookieAuthenticationDefaults.AuthenticationScheme);

ClaimsPrincipal user = **new** ClaimsPrincipal(claimsIdentity);

//登入使用者，相當於ASP.NET中的FormsAuthentication.SetAuthCookie

HttpContext.SignInAsync(CookieAuthenticationDefaults.AuthenticationScheme, user).Wait();

//可以使用HttpContext.SignInAsync方法的過載來定義持久化cookie儲存使用者認證資訊，例如下面的程式碼就定義了使用者登入後60分鐘內cookie都會保留在客戶端計算機硬碟上，

//即便使用者關閉了瀏覽器，60分鐘內再次訪問站點仍然是處於登入狀態，除非呼叫Logout方法登出登入。

/\*

HttpContext.SignInAsync(

CookieAuthenticationDefaults.AuthenticationScheme,

user, new AuthenticationProperties() { IsPersistent = true, ExpiresUtc = DateTimeOffset.Now.AddMinutes(60) }).Wait();

\*/

**return** View();

}

3.讀取登入使用者資訊

那麼使用者登入後怎麼將登入使用者的資訊（比如使用者名稱）讀取出來呢？我們在HomeController的Index方法中演示瞭如何判斷當前使用者是否已經登入，並且讀出登入使用者的使用者名稱，Index方法的程式碼如下所示：

**///** **<summary>**

**///** 該Action判斷使用者是否已經登入，如果已經登入，那麼讀取登入使用者的使用者名稱

**///** **</summary>**

**public** IActionResult **Index**()

{

//如果HttpContext.User.Identity.IsAuthenticated為true，

//或者HttpContext.User.Claims.Count()大於0表示使用者已經登入

**if** (HttpContext.User.Identity.IsAuthenticated)

{

//這裡通過 HttpContext.User.Claims 可以將我們在Login這個Action中儲存到cookie中的所有

//claims鍵值對都讀出來，比如我們剛才定義的UserName的值Wangdacui就在這裡讀取出來了

**var** userName = HttpContext.User.Claims.First().Value;

}

**return** View();

}

4.登出使用者

那麼登入使用者後怎麼登出登入呢？我們在HomeController的Logout方法中演示瞭如何登出登入的使用者，程式碼如下所示：

**///** **<summary>**

**///** 該Action從Asp.Net Core中登出登入的使用者

**///** **</summary>**

**public** IActionResult **Logout**()

{

//登出登入的使用者，相當於ASP.NET中的FormsAuthentication.SignOut

HttpContext.SignOutAsync().Wait();

**return** View();

}

前面說了實際上在ASP.NET CORE的Cookie認證中還可以設定Cookie的名字、是否持久化儲存等。

以上就是本次小編整理的全部內容，感謝你對指令碼之家的支援。