عنوان: **فراخوانی متدهای Controllerها در Viewهای ASP.NET MVC**
نویسنده: تایماز قبادی
تاریخ: ۱۳۹۳/۱۱/۰۶ ۱۵:۱۵
آدرس: www.dotnettips.info
گروه‌ها: C#, MVC, Delegate, Func

شاید در ابتدا فراخوانی متدی از یک کنترلر در یک View کار سختی به نظر برسد، ولی در واقع با استفاده از مفاهیم Lambda expressions و Delegateها این کار بسیار راحت خواهد بود.

برای این کار میتوانیم متد مورد نظر را به صورت یک delegate تعریف کرده و به view ارسال کنیم. فرض کنیم متدی داریم برای برگرداندن مجموع 2 عدد به صورت string:

```
public string Sum(int a,int b)
{
    return (a + b).ToString();
}
```

حال برای اینکه بتوانیم این متد را بصورت یک delegate به view ارسال کنیم لازم است تا یک delegate را بصورت public و در خارج از تعریف کلاسها و درون یک namespace مشخصی تعریف کنیم. در اینجا برای راحتی در همان MvcTest.Controllers namespace یک delegate را بصورت زیر (MvcTest نام پروژه است) تعریف می‌کنیم:

```
public delegate string SumOf2Number(int a, int b);
```

حال میتوانیم بصورت زیر این متد را از طریق ViewBag به View ارسال کنیم:

```
SumOf2Number sum2numbers = Sum;
ViewBag.SumFunc3 = sum2numbers;
```

در روش دوم، می‌توانیم متد مورد نظر را بصورت Func به View ارسال کنیم. این کار را میتوانیم به دو صورت انجام دهیم، که هر دو را در تکه کد زیر خواهید دید:

```
ViewBag.SumFunc =(Func<int,int,string>) Sum;//way 1
ViewBag.SumFunc2 = (Func<int, int, string>)((int a, int b) => { return (a + b).ToString(); });//way 2
```

همانطور که متوجه شدید، در روش اول تنها کاری که کردیم متد Sum را از طریق TypeCasting به یک Func تبدیل کردیم و در روش دوم هم یک Lambda expression را بصورت مستقیم به Func تبدیل کرده و استفاده کردیم.

میتوانیم یک Lambda expression را به یک متغیر delegate نیز ربط دهیم؛ به این صورت:

```
SumOf2Number sum2numbers2 = (int a, int b) => { return (a + b).ToString(); };
```

در نهایت کد بخش کنترلر کلاً به اینصورت خواهد بود:

```
namespace MvcTest.Controllers
{
    public  delegate string SumOf2Number(int a, int b);
```

```
    public class HomeController : Controller
    {
        public ActionResult Index()
        {
            SumOf2Number sum2numbers = Sum;
            SumOf2Number sum2numbers2 = (int a, int b) => { return (a + b).ToString(); };

            ViewBag.SumFunc =(Func<int,int,string>) Sum;
            ViewBag.SumFunc2 = (Func<int, int, string>)((int a, int b) => { return (a + b).ToString();
});
            ViewBag.SumFunc3 = sum2numbers;
            ViewBag.SumFunc4 = sum2numbers2;
            return View();
        }
        public string Sum(int a,int b)
        {
            return (a + b).ToString();
        }
        public ActionResult About()
        {
            ViewBag.Message = "Your application description page.";

            return View();
        }

        public ActionResult Contact()
        {
            ViewBag.Message = "Your contact page.";

            return View();
        }
    }
}
```

و در Index View خواهیم داشت: (البته اصولاً استفاده از controller namespace در سمت view کار درستی نیست، منتها اینجا فقط یک مثال کاربردی ساده است)

```
@using MvcTest.Controllers;
@{
    ViewBag.Title = "Home Page";
}

<h1>
    @ViewBag.SumFunc(7,8)
</h1>
<h1>
    @ViewBag.SumFunc2(9, 10)
</h1>
<h1>
    @ViewBag.SumFunc3(5, 1)
</h1>
<h1>
    @ViewBag.SumFunc4(2, 3)
</h1>
```