عنوان: **Cookie - قسمت سوم**
نویسنده: یوسف نژاد
تاریخ: ۱۳۹۲/۰۳/۳۱ ۱:۳۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Cookie, Http Cookie

Cookie - قسمت اول : مقدمه، تاریخچه، معرفی، و شرح کامل

Cookie - قسمت دوم : کوکی در جاوا اسکریپت

نکته مهم: خواندن قسمت‌های قبلی این سری (مخصوصا قسمت اول) برای درک بهتر مطالب پیشنهاد می‌شود.

.

کوکی در ASP.NET - بخش اول

در قسمت‌های قبلی مقدمات و مباحث کلی راجع به کوکی‌ها و انواع آن، شرح کامل خواص، نحوه رفتار مرورگرها با انواع کوکی‌ها و درنهایت نحوه کار کردن با کوکی‌ها در سمت کلاینت با استفاده از زبان محبوب جاوا اسکریپت پرداخته شد.

در ادامه این سری مطالب به نحوه برخورد ASP.NET با کوکی‌ها و چگونگی کار کردن با کوکی در سمت سرور آشنا خواهیم شد. در بخش اول این قسمت مباحث ابتدایی و اولیه برای کار با کوکی‌ها در ASP.NET ارائه می‌شود. در بخش دوم مباحث پیشرفته‌تر همچون SubCookieها در ASP.NET و نیز سایر نکات ریز کار با کوکی‌ها در ASP.NET بحث خواهد شد.

.

Response و Request در ASP.NET

در قسمت اول این سری به مفاهیم Http Response و Http Request اشاره کوتاهی شده بود. به‌صورت خلاصه، درخواستی که از

سمت یک کلاینت به یک وب سرور ارسال می‌شود **Request** و پاسخی که وب سرور به آن درخواست می‌دهد **Response** نامیده می‌شود.

در ASP.NET، کلیه اطلاعات مرتبط با **درخواست** رسیده از سمت یک کلاینت در نمونه‌ای منحصر به فرد از کلاس HttpRequest نگه‌داری می‌شود. محل اصلی نگه‌داری این نمونه در پراپرتی Request از نمونه جاری کلاس System.Web.HttpContext (قابل دسترسی ازطریق HttpContext.Current) است. البته کلاس Page هم یک پراپرتی با نام Request دارد که دقیقا از همین پراپرتی کلاس HttpContext استفاده می‌کند.

هم‌چنین کلیه اطلاعات مرتبط با **پاسخ** ارسالی وب سرور به سمت کلاینت مربوطه در نمونه‌ای از کلاس HttpResponse ذخیره می‌شود. محل اصلی نگه‌داری این نمونه نیز در پراپرتی Response از نمونه جاری کلاس HttpContext است. همانند Request، کلاس Page یک پراپرتی با نام Response برای نگه‌داری این نمونه دارد که این هم دقیقا از پراپرتی متناظر در کلاس HttpContext استفاده می‌کند.

.

.

کوکی‌ها در Response و Request

هر دو کلاس HttpResponse و HttpRequest یک پراپرتی با عنوان Cookies (^ و ^) دارند که مخصوص نگهداری کوکی‌های مربوطه هستند. این پراپرتی از نوع System.Web.HttpCookieCollection است که یک کالکشن مخصوص برای ذخیره کوکی‌هاست.

- این پراپرتی (Cookies) در کلاس HttpRequest محل نگه‌داری کوکی‌های ارسالی توسط مرورگر در درخواست متناظر آن است. کوکی‌هایی که مرورگر با توجه به شرایط جاری و تنظیمات کوکی‌ها اجازه ارسال به سمت سرور را به آن‌ها داده و در درخواست ارسالی ضمیمه کرده است (با استفاده از هدر :Cookie که در قسمت اول شرح داده شد) و ASP.NET پس از پردازش و Parse داده‌ها، درون این پراپرتی اضافه کرده است.

- این پراپرتی (Cookies) در کلاس HttpResponse محل ذخیره کوکی‌های ارسالی از وب سرور به سمت مرورگر کلاینت در پاسخ به درخواست متناظر است. کوکی‌های درون این پراپرتی پس از بررسی و استخراج داده‌های موردنیاز توسط ASP.NET در هدر پاسخ ارسالی ضمیمه خواهند شد (با استفاده از هدر :Set-Cookie که در قسمت اول توضیح داده شد).

.

.

ایجاد و به‌روزرسانی کوکی در ASP.NET

برای ایجاد یک کوکی و ارسال آن به سمت کلاینت همان‌طور که در بالا نیز اشاره شد، باید از پراپرتی Cookies از کلاس HttpContext استفاده کرد. برای ایجاد یک کوکی روش‌های مختلفی وجود دارد.
در روش اول با استفاده از ویژگی مخصوص **ایندکسر** کلاس HttpCookieCollection عملیات تولید کوکی انجام می‌شود. در این روش، ابتدا بررسی می‌شود که کوکی موردنظر در لیست کوکی‌های جاری وجود دارد یا خیر. درصورتی‌که با این نام قبلا یک کوکی ثبت شده باشد، مقدار کوکی موجود بروزرسانی خواهد شد. اما اگر این نام وجود نداشته باشد یک کوکی جدید با این نام به لیست افزوده شده و مقدار آن ثبت می‌شود. مثال:

```
HttpContext.Current.Response.Cookies["myCookie"].Value = "myCookieValue";
```

روش بعدی استفاده از متد Add در کلاس HttpCookieCollection است. در این روش ابتدا یک نمونه از کلاس HttpCookie ایجاد شده و سپس این نمونه به لیست کوکی‌ها اضافه می‌شود. کد زیر چگونگی استفاده از این روش را نشان می‌دهد:

```
var myCookie = new HttpCookie("myCookie", "myCookieValue");
HttpContext.Current.Response.Cookies.Add(myCookie);
```

روش دیگر استفاده از متد Set کلاس HttpCookieCollection است. تفاوت این متد با متد Add در این است که متد Set ابتدا سعی می‌کند عملیات update انجام دهد. یعنی عملیات افزودن تنها وقتی‌که نام کوکی موردنظر در لیست کوکی‌ها یافته نشود انجام خواهد شد. برای مثال:

```
HttpContext.Current.Response.Cookies.Set(new HttpCookie("myCookie", "myCookieValue"));
```

نکته: باتوجه به توضیحات بالا، متد Set اجازه افزودن دو کوکی با یک نام را نمی‌دهد. برای اینکار باید از متد Add استفاده کرد. درباره این موضوع در قسمت بعدی بیشتر توضیح داده خواهد شد.
روش دیگری که برای ایجاد یکی کوکی می‌توان از آن استفاده کرد، بکارگیری متد AppnedCookie از کلاس HttpResponse است. در این روش نیز ابتدا باید یک نمونه از کلاس HttpCookie تولید شود. این روش همانند استفاده از متد Add از کلاس HttpCookieCollection است. کد زیر مثالی از این روش را نشان می‌دهد:

```
HttpContext.Current.Response.AppendCookie(new HttpCookie("myCookie", "myCookieValue"));
```

روش بعدی استفاده از متد SetCookie از کلاس HttpResponse است. فرق این متد با متد AppendCookie در این است که در متد SetCookie ابتدا وجود یک کوکی با نام ارائه شده بررسی می‌شود و درصورت وجود، مقدار این کوکی بروزرسانی می‌شود. درصورتی‌که قبلا یک کوکی با این نام وجود نداشته باشد، یک کوکی جدید به لیست کوکی‌ها اضافه می‌شود. این روش همانند استفاده از متد Set از کلاس HttpCookieCollection است. نمونه‌ای از نحوه استفاده از این متد در زیر آورده شده است:

```
HttpContext.Current.Response.SetCookie(new HttpCookie("myCookie", "myCookieValue"));
```

نکته: تمامی فرایندهای نشان داده شده در بالا تنها موجب تغییر محتویات کالکشن کوکی‌ها درون HttpContext می‌شود و تا زمانی‌که توسط وب سرور با استفاده از دستور Set-Cookie به سمت مرورگر ارسال نشوند تغییری در کلاینت بوجود نخواهند آورد.

برای آشنایی بیشتر با این روند کد زیر را برای تعریف یک کوکی جدید درنظر بگیرید:

```
HttpContext.Current.Response.Cookies["myCookie"].Value = "myValue";
```

برای مشاهده هدر تولیدی توسط وب سرور می‌توان از نرم افزار محبوب Fiddler استفاده کرد (از اواخر سال 2012 که نویسنده این ابزار به Telerik پیوسته، توسعه آن بسیار فعال‌تر شده و نسخه‌های جدید با لوگوی جدید! ارائه شده است).
تصویر زیر مربوط به مثال بالاست:

همانطور که مشاهده می‌کنید دستور ایجاد یک کوکی با نام و مقدار وارده در هدر پاسخ تولیدی توسط وب سرور گنجانیده شده است.

نکته: در ASP.NET به صورت پیش فرض از مقدار "/" برای پراپرتی Path استفاده می‌شود.

.

خواص کوکی در ASP.NET

برای تعیین یا تغییر خواص یک کوکی در ASP.NET باید به نمونه HttpCookie مربوطه دست یافت. سپس با استفاده از پراپرتی‌های این کلاس می‌توان خواص موردنظر را تعیین کرد. برای مثال:

```
var myCookie = new HttpCookie(string.Empty);
myCookie.Name = "myCookie";
myCookie.Value = "myCookieValue";
myCookie.Domain = "dotnettip.info";
myCookie.Path = "/post";
myCookie.Expires = new DateTime(2015, 1, 1);
myCookie.Secure = true;
myCookie.HttpOnly = true;
```

نکته مهم: امکان تغییر خواص یک کوکی به صورت مستقیم در سمت سرور وجود ندارد. درواقع برای اعمال این تغییرات در سمت کلاینت باید به ازای هر کوکی موردنظر یک کوکی جدید با مقادیر جدید ایجاد و به کالکشن کوکی‌ها در Http Response مربوطه اضافه شود تا پس از قرار دادن دستور Set-Cookie متناظر در هدر پاسخ ارسالی به سمت کلاینت و اجرای آن توسط مرورگر، مقادیر خواص مورنظر در سمت کلاینت بروزرسانی شوند. دقت کنید که تمامی نکات مرتبط با هویت یک کوکی که در قسمت اول شرح داده شد در اینجا نیز کاملا صادق است.

روش دیگری نیز برای تعیین **برخی** خواص کوکی‌ها به صورت کلی در فایل وب کانفیگ وجود دارد. برای اینکار از تگ httpCookies

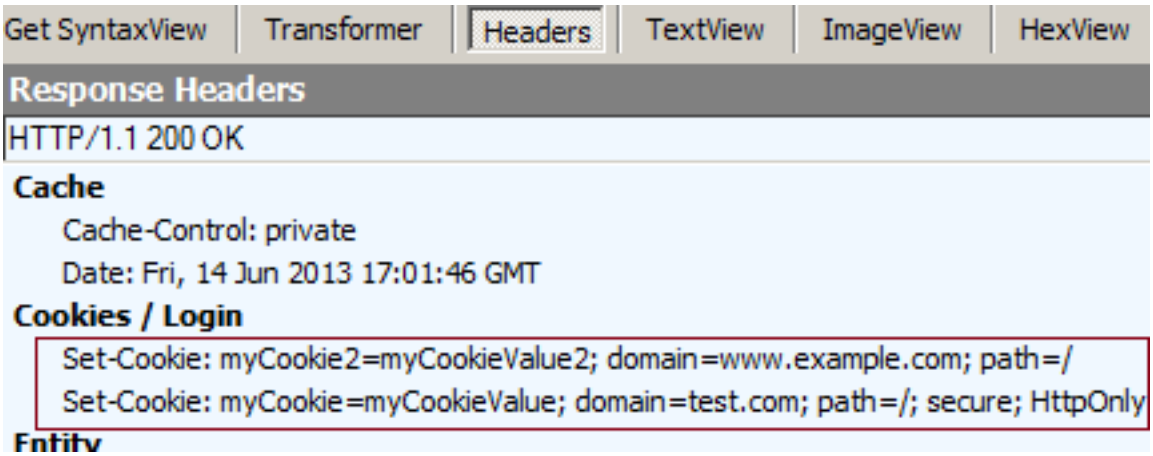
در قسمت system.web استفاده می‌شود. برای مثال:

```
<httpCookies domain="www.example.com" httpOnlyCookies="true" requireSSL="true" />
```

این امکان از ASP.NET 2.0 به بعد اضافه شده است. با استفاده از این تگ، تنظیمات اعمال شده برای تمامی کوکی‌ها درنظر گرفته می‌شود. البته درصورتی‌که تنظیم موردنظر برای کوکی به صورت صریح آورده نشده باشد. برای نمونه به کد زیر دقت کنید:

```
var myCookie = new HttpCookie("myCookie", "myCookieValue");
myCookie.Domain = "test.com";
HttpContext.Current.Response.Cookies.Add(myCookie);
var myCookie2 = new HttpCookie("myCookie2", "myCookieValue2");
myCookie2.HttpOnly = false;
myCookie2.Secure = false;
HttpContext.Current.Response.Cookies.Add(myCookie2);
```

با استفاده از تنظیمات تگ httpCookies که در بالا نشان داده شده است، هدر پاسخ تولیدی توسط وب سرور به صورت زیر خواهد بود:



همان‌طور که می‌بینید تنها مقادیر پراپرتی‌هایی که صراحتا برای کوکی آورده نشده است از تنظیمات وب کانفیگ خوانده می‌شود.

.

.

حذف کوکی در ASP.NET

برای حذف یک کوکی در ASP.NET یک روش کلی وجود دارد که در قسمت‌های قبلی نیز شرح داده شده است، یعنی تغییر خاصیت Expires کوکی به تاریخی در گذشته. برای نمونه داریم:

```
var myCookie = new HttpCookie("myCookie", "myCookieValue");
myCookie.Expires = DateTime.Now.AddYears(-1);
```

نکته مهم: در کلاس HttpCookieCollection یک متد با نام Remove وجود دارد. از این متد برای حذف یک کوکی از لیست موجود

در این کلاس استفاده می‌شود. دقت کنید که حذف یک کوکی از لیست کوکی‌ها با استفاده از این متد تاثیری بر موجودیت آن کوکی در سمت کلاینت نخواهد گذاشت و تنها روش موجود برای حذف یک کوکی در سمت کلاینت همان تنظیم مقدار خاصیت Expires است.

.

خواندن کوکی در ASP.NET

برای خواندن مقدار یک کوکی ارسالی از مرورگر کلاینت در ASP.NET، باتوجه به توضیحات ابتدای این مطلب، طبیعی است که باید از پراپرتی **Request.** Cookies در نمونه جاری از کلاس HttpContext استفاده کرد. برای این کار نیز چند روش وجود دارد.
روش اول استفاده از **ایندکسر** کلاس HttpCookieCollection است. برای اینکار نیاز به نام یا ایندکس کوکی موردنظر در لیست مربوطه داریم. برای مثال:

```
var myCookie = HttpContext.Current.Request.Cookies["myCookie"];
```

یا این نمونه با استفاده از **ایندکسر عددی** :

```
var myCookie = HttpContext.Current.Request.Cookies[0];
```

روش دیگری که برای خواند مقدار یک کوکی می‌توان بکار برد، استفاده از متد Get از کلاس HttpCookieCollection است. این متد همانند ایندکسر این کلاس نیاز به نام یا ایندکس کوکی موردنظر دارد. برای نمونه:

```
var myCookie = HttpContext.Current.Request.Cookies.Get("myCookie");
```

یا استفاده از ایندکس کوکی :

```
var myCookie = HttpContext.Current.Request.Cookies.Get(0);
```

.

.

بحث و نتیجه گیری

تا اینجا با مفاهیم اولیه درباره نحوه برخورد ASP.NET با کوکی‌ها آشنا شدیم. روش‌های مختلف ایجاد و یا به‌روزرسانی کوکی‌ها نشان داده شد. با تعیین انواع خواص کوکی‌ها آشنا شدیم. نحوه حذف یک کوکی در ASP.NET را دیدیم. روش‌های خواندن مقادیر کوکی‌ها را نیز مشاهده کردیم.

باز هم تاکید می‌کنم که تمامی تغییرات اعمالی در سمت سرور تا زمانی‌که به‌صورت دستورات Set-Cookie در هدر پاسخ وب سرور قرار نگیرند هیچ کاری در سمت کلاینت انجام نمی‌دهند.

در قسمت بعدی این سری مطالب به مباحث پیشرفته‌تری چون SubCookieها در ASP.NET و هویت منحصر به فرد کوکی‌ها در سمت سرور پرداخته می‌شود.

.

.

منابع

http://msdn.microsoft.com/en-us/library/ms178194(v=vs.100).aspx

http://msdn.microsoft.com/en-us/library/aa289495(v=vs.71).aspx

http://www.codeproject.com/Articles/31914/Beginner-s-Guide-To-ASP-NET-Cookies

http://www.codeproject.com/Articles/244904/Cookies-in-ASP-NET

نظرات خوانندگان

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۳/۰۸/۰۶ ۱۸:۱۸

یک کوکی ایجاد کردم و یه سری دیتا درون اون قرار دادم. در جایی از سایت برای پرداخت اینترنتی به درگاه بانک متصل میشم اما بعد از برگشت از بانک ، وقتی میخوام مقدار کوکی رو بخونم ، کوکی مقدارش خالیه. حتی دستور Page.User.Identity.IsAuthenticate هم مقدار false بر می‌گردونه.در صورتی که قبل از ارسال به درگاه، کاربر لاگین بوده.
ممنون میشم راهنمایی کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۰:۹

در دات نت 4.5 ، مشکل طولانی بودن حاصل BinaryFormatter serialization برطرف شده (January 2013). این مشکل سبب می‌شده تا حاصل RolePrincipal.ToEncryptedTicket بسیار طولانی شده و بیشتر از حد مجاز اندازه قابل ذخیره سازی در یک کوکی شود.
- وصله‌ی نسخه‌ی ویندوز 8 و ویندوز سرور 2012 آن از اینجا قابل دریافت است؛ نسخه‌ی ویندوز 7 و ویندوز سرور 2008 از اینجا .
+ آپدیت ویندوز را روشن کنید تا آخرین به روز رسانی‌ها و نگارش‌های دات نت نصب شده را به صورت خودکار دریافت کنید.

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۲:۳۰

با تشکر؛ فقط یک نکته تکمیلی که فراموش کردم اینکه مشکل مورد نظر مربوط به asp.net web form هست و من از simple membership برای فرآیند احراز هویت استفاده میکنم.با توجه به این مسائل ، انجام مواردی که شما فرمودید برای حل مشکل کفایت میکنه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۲:۳۵

فرقی نمی‌کند. مباحث پایه Forms Authentication برای تمام فناوری‌هایی که از آن استفاده می‌کنند یکسان است.