

## مقدمه

OutputCaching باعث می‌شود خروجی یک اکشن متد در حافظه نگهداری شود. با اعمال این نوع کشینگ، ASP.NET در خواست‌های بعدی به این اکشن را تنها با بازگرداندن همان مقدار قبلی نگهداری شده در کش، پاسخ می‌دهد. در حقیقت با OutputCaching از تکرار چند باره کد درون یک اکشن در فراخوانی‌های مختلف جلوگیری کرده‌ایم. کش کردن باعث می‌شود که کارایی و سرعت سایت افزایش یابد؛ اما باید دقت کنیم که چه موقع و چرا از کش کردن استفاده می‌کنیم و چه موقع باید از این کار امتناع کرد.

## فواید کش کردن

- انجام عملیات هزینه دار فقط یکبار صورت می‌گیرد. (هزینه از لحاظ فشار روی حافظه سرور و کاهش سرعت بالا آمدن سایت)
- بار روی سرور در زمان‌های پیک کاهش می‌یابد.
- سرعت بالا آمدن سایت بیشتر می‌شود.

## چه زمانی باید کش کرد؟

- وقتی محتوای نمایشی برای همه کاربران یکسان است.
- وقتی محتوای نمایشی برای نمایش داده شدن، فشار زیادی روی سرور تحمیل می‌کند.
- وقتی محتوای نمایشی به شکل مکرر در طول روز باید نمایش داده شود.
- وقتی محتوای نمایشی به طور مکرر آپدیت نمی‌شود. (در مورد تعریف کیفیت "مکرر"، برنامه نویسی بهترین تصمیم گیرنده است)

## طرح مساله

فرض کنید صفحه اول سایت شما دارای بخش‌های زیر است :

خلاصه اخبار بخش علمی، خلاصه اخبار بخش فرهنگی ، ده کامنت آخر، لیستی از کتگوری‌های موجود در سایت.

روش‌های مختلفی برای کوئری گرفتن وجود دارد، به عنوان مثال ما به کمک یک یا چند کوئری و توسط یک ViewModel جامع، می‌خواهیم اطلاعات را به سمت ویو ارسال کنیم. پس در اکشن متد Index ، حجم تقریباً کمی از اطلاعات را باید به کمک کوئری(کوئری‌های) تقریباً پیچیده ای دریافت کنیم و اینکار به ازای هر ریکوئست هزینه دارد و فشار به سرور وارد خواهد شد. از طرفی می‌دانیم صفحه اول ممکن است در طول یک یا چند روز تغییر نکند و همچنین شاید در طول یکساعت چند بار تغییر کند! به هر حال در جایی از سایت قرار داریم که کوئری (کوئری‌های) مورد نظر زیاد صدا زده میشوند ، در حقیقت صفحه اول احتمالاً بیشترین فشار ترافیکی را در بین صفحات ما دارد، البته این فقط یک احتمال است و ما دقیقاً از این موضوع اطلاع نداریم.

یکی از راه‌های انجام یک کش موفق و دانستن لزوم کش کردن، این است که دقیقاً بدانیم ترافیک سایت روی چه صفحه ای بیشتر است. در واقع باید بدانیم در کدام صفحه "هزینه‌ی اجرای عملیات موجود در کد" بیشترین است.

فشار ترافیکی (ریکوئست‌های زیاد) و آپدیت‌های روزانه‌ی دیتابیس را، در دو کفه ترازو قرار دهید؛ چه کار باید کرد؟ این تصمیمی است که شما باید بگیرید. نگرانی خود را در زمینه آپدیت‌های روزانه و ساعتی کمتر کنید؛ در ادامه راهی را معرفی میکنیم که آپدیت‌های هر از گاه شما، در پاسخ ریکوئست‌ها دیده شوند. کمی کفهی کش کردن را سنگین کنید.

به هر حال، فعال کردن قابلیت کش کردن برای یک اکشن، بسیار ساده است، کافیت ویژگی (attribute) آن را بالای اکشن بنویسید :

```
[OutputCache(Duration = "60", Location = OutputCacheLocation.Server)]
public ActionResult Index()
{
    // کوثری یا کوثری‌های لازم برای استفاده در صفحه اصلی و تبدیل آن به یک ویو مدل جامع//
}
```

```
[OutputCache(CacheProfile = "FirstPageIndex", Location=OutputCacheLocation.Server)]
public ActionResult Index()
{
    // کوثری یا کوثری‌های لازم برای استفاده در صفحه اصلی و تبدیل آن به یک ویو مدل جامع//
}
```

دو روش فوق برای کش کردن خروجی Index از لحاظ عملکرد یکسان است، به شرطی که در حالت دوم در وب کانفیگ و در بخش system.web آن، یک پروفایل ایجاد کنیم کنیم :

```
<caching>
  <outputCacheSettings>
    <outputCacheProfiles>
      <add name="FirstPageIndex" duration="60"/>
    </outputCacheProfiles>
  </outputCacheSettings>
</caching>
```

در حالت دوم ما یک پروفایل برای کشینگ ساخته ایم و در ویژگی بالای اکشن متد، آن پروفایل را صدا زده ایم. از لحاظ منطقی در حالت دوم، چون امکان استفاده مکرر از یک پروفایل در جاهای مختلف فراهم شده، روش بهتری است. محل ذخیره کش نیز در هر دو حالت سرور تعریف شده است.

برای تست عملیات کشینگ، کافیت یک BreakPoint درون Index قرار دهید و برنامه را اجرا کنید. پس از اجرا، برنامه روی Break Point می‌ایستد و اگر F5 را بزنی، سایت بالا می‌آید. بار دیگر صفحه را رفرش کنیم، **اگر این "بار دیگر" در کمتر از 60 ثانیه پس از رفرش قبلی اتفاق افتاده باشد برنامه روی Break Point متوقف نخواهد شد**، چون خروجی اکشن، در کش بر روی سرور ذخیره شده است و این یعنی ما فشار کمتری به سرور تحمیل کرده ایم، صفحه با سرعت بالاتری در دسترس خواهد بود.

ما از تکرار اجرای کد جلوگیری کرده ایم و عدم اجرای کد بهترین نوع بهینه سازی برای یک سایت است. [اسکات الن، پلورال سایت]

### چطور زمان مناسب برای کش کردن یک اکشن را انتخاب کنیم؟

- **کشینگ با زمان کوتاه** ؛ فرض کنید زمان کش را روی 1 ثانیه تنظیم کرده اید. این یعنی اگر ریکوئست هایی به یک اکشن ارسال شود و همه در طول یک ثانیه اتفاق بیفتد، آن اکشن فقط برای بار اول اجرا میشود، و در بارهای بعد(در طول یک ثانیه) فقط محتوای ذخیره شده در آن یک اجرا، بدون اجرای جدید، نمایش داده میشود. پس سرور شما فقط به یک ریکوئست در ثانیه در طول روز جواب خواهد داد و ریکوئست‌های تقریباً همزمان دیگر، در طول همان ثانیه، از نتایج آن ریکوئست (اگر موجود باشد) استفاده خواهند کرد

- **کشینگ با زمان طولانی** ؛ ما در حقیقت با اینکار منابع سرور را حفاظت میکنیم، چون عملیات هزینه دار(مثل کوثری‌های حجیم) تنها یکبار در طول زمان کشینگ اجرا خواهند شد. مثلاً اگر تنظیم زمان روی عدد 86400 تنظیم شود(یک روز کامل)، پس از اولین

ریکوئست به اکشن مورد نظر، تا 24 ساعت بعد، این اکشن اجرا نخواهد شد و فقط خروجی آن نمایش داده خواهد شد. آیا دلیلی دارد که یک کوئری هزینه دار را که قرار نیست خروجی اش در طول روز تغییر کند به ازای هر ریکوئست یک بار اجرا کنیم؟

### اگر اطلاعات موجود در دیتابیس را تغییر دهیم چه کار کنیم که کشینگ رفرش شود؟

فرض کنید در همان مثال ابتدای این مقاله، شما یک پست به دیتابیس اضافه کرده اید، اما چون مثلاً duration مربوط به کشینگ را روی 86400 تعریف کرده اید تا 24 ساعت از زمان ریکوئست اولیه نگذرد، سایت آپدیت نخواهد شد و محتوا همان چیزهای قبلی باقی خواهند ماند. اما چاره چیست؟

کافیست در بخش ادمین، وقتی که یک پست ایجاد میکنید یا پستی را ویرایش میکند در اکشن‌های مرتبط با Create یا Edit یا Delete چنین کدی را پس از فرمان ذخیره تغییرات در دیتابیس، بنویسید:

```
Response.RemoveOutputCacheItem(Url.Action("index", "home"));
```

واضح است که ما داریم کشینگ مرتبط با یک اکشن متد مشخص را پاک میکنیم. با اینکار در اولین ریکوئست پس از تغییرات اعمال شده در دیتابیس، ASP.NET MVC چون میبند گشی برای این اکشن وجود ندارد، متد را اجرا میکند و کوئری‌های درونش را خواهد دید و اولین ریکوئست پیش از گش شدن را انجام خواهد داد. با اینکار کشینگ ریست شده است و پس از این ریکوئست و استخراج اطلاعات جدید، زمان کشینگ صفر شده و آغاز میشود.

میتوانید یک دکمه در بخش ادمین سایت طراحی کنید که هر موقع دلتان خواست کلیه کش‌ها را به روش فوق پاک کنید! تا اپلیکیشن منتظر ریکوئست‌های جدید بماند و کش‌ها دوباره ایجاد شوند.

### جمع بندی

ویژگی OutputCach دارای پارامترهای زیادیست و در این مقاله فقط به توضیح عملکرد این اتریبیوت اکتفا شده است. بطور کلی این مبحث ظاهر ساده ای دارد، ولی نحوه استفاده از کشینگ کاملاً وابسته به هوش برنامه نویسی است و پیچیدگی‌های مرتبط با خود را دارد. در واقع خیلی مشکل است که بتوانید یک زمان مناسب برای کش کردن تعیین کنید. باید برنامه خود را در یک محیط شبیه سازی تحت بار قرار دهید و به کمک اندازه گیری و محاسبه به یک قضاوت درست از میزان زمان کش دست پیدا کنید. گاهی متوجه خواهید شد، از مقدار زیادی از حافظه سیستم برای کش کردن استفاده کرده اید و در حقیقت آنقدر ریکوئست ندارید که احتیاج به این هزینه کردن باشد.

یکی از روش‌های موثر برای دستیابی به زمان بهینه برای کش کردن استفاده از CacheProfile درون وب کانفیگ است. وقتی از کشینگ استفاده میکنید، در همان ابتدا مقدار زمانی مشخص برای آن در نظر نگرفته اید (در حقیقت مقدار زمان مشخصی نمیدانید) پس مجبور به آزمون و خطا و تست و اندازه گیری هستید تا بدانید چه مقدار زمانی را برای چه پروفایلی قرار دهید. مثلاً پروفایل هایی به شکل زیر تعریف کرده اید و نام آنها را به اکشن‌های مختلف نسبت داده اید. به راحتی میتوانید از طریق دستکاری وب کانفیگ مقادیر آن را تغییر دهید تا به حالت بهینه برسید، بدون آنکه کد خود را دستکاری کنید.

```
<caching>
  <outputCacheSettings>
    <outputCacheProfiles>
      <add name="Long" duration="86400"/>
      <add name="Average" duration="43600"/>
      <add name="Short" duration="600"/>
    </outputCacheProfiles>
  </outputCacheSettings>
</caching>
```

برای مطالعه جزئیات بیشتر در مورد OutputCaching مقالات زیر منابع مناسبی هستند.

## نظرات خوانندگان

نویسنده: ابوالفضل رجب پور  
تاریخ: ۹:۵۳ ۱۳۹۳/۰۴/۲۹

سلام

یک ابهام در یک مثال واقعی مثلا سایت خبری.

اگر بخواهیم خروجی اکشن اخبار رو کش کنیم، و در عین حال تعداد بازدید از هر خبر رو هم ثبت کنیم، چطور باید این کار رو انجام داد؟

نویسنده: مرتضی دلیل  
تاریخ: ۱۰:۲۴ ۱۳۹۳/۰۴/۲۹

قاعدتا اگر اکشن مربوط به نمایش هر خبر مستقل از اکشن نمایش "آخرین اخبار" باشد، با کش کردن اکشن "آخرین اخبار" مشکلی برای اکشن نمایش دهنده هر خبر بوجود نخواهد آمد و میتوان در این اکشن، متدها یا عملیات مورد نظر را بدون نگرانی اعمال کرد. (اگر منظور از "ثبت"، ذخیره‌ی اطلاعات باشد)

نویسنده: وحید نصیری  
تاریخ: ۱۰:۲۶ ۱۳۹۳/۰۴/۲۹

- با استفاده از jQuery که یک بحث سمت کاربر است، زمانیکه صفحه نمایش داده شد، یک درخواست Ajax ایی به اکشن متدی خاص، جهت به روز رسانی تعداد بار مشاهده ارسال کنید. به این روش client side tracking هم می‌گویند (کل اساس کار Google analytics به همین نحو است).

- روش دوم استفاده از [Donut Caching](#) است. در یک چنین حالتی، کد زیر مجاز است:

```
[LogThis]
[DonutOutputCache(Duration=5, Order=100)]
public ActionResult Index()
```

[اطلاعات بیشتر](#)

نویسنده: ایلیا اکبری فرد  
تاریخ: ۱۶:۵۲ ۱۳۹۳/۰۸/۱۲

با سلام.

متدی به روش زیر در کنترلر خود ایجاد کرده ام:

```
[OutputCache(Duration = (7 * 24 * 60 * 60), VaryByParam = "none")]
[AllowAnonymous]
public virtual ActionResult Notification()
{
    ....
}
```

و در قسمت ادمین سیستم که در یک area جداگانه قرار دارد در اکشن متد خود اینگونه نوشتم:

```
Response.RemoveOutputCacheItem(Url.Action("Notification", "Article"));
Response.RemoveOutputCacheItem(Url.Action("Notification", "Article", new { area = "" }));
```

هیچکدام از دو روش بالا برایم جواب نمی‌دهد و کش خالی نمی‌شود. علت چیست؟

نویسنده: محسن خان  
تاریخ: ۱۸:۴۱ ۱۳۹۳/۰۸/۱۲

آیا از `Html.RenderAction` برای نمایش آن استفاده کردید؟ اگر بله، متد یاد شده تاثیری روی کش آن نداره، چون نحوه‌ی کش شدن `child action`ها متفاوت.

نویسنده: ایلیا اکبری فرد  
تاریخ: ۱۹:۱۵ ۱۳۹۳/۰۸/۱۲

بله. راه حل مشکل چیست؟

نویسنده: وحید نصیری  
تاریخ: ۱۵:۱۴ ۱۳۹۳/۰۸/۱۳

به این صورت؛ البته این روش کش تمام `child action`ها را با هم پاک می‌کند:

```
OutputCacheAttribute.ChildActionCache = new MemoryCache("NewRandomStringNameToClearTheCache");
```