در این مقاله قصد داریم که چرخه حیات برنامه mvc از زمان دریافت یک درخواست تا ارائه پاسخ به آن را شرح دهیم .کلا دو چرخه حیات مهم در mvc داریم که یکی Application و دیگری Request است .چرخه حیات application زمان شروع اجرای برنامه در سرور تا آخر است و چرخه حیات درخواست در واقع از زمان شروع یک درخواست تا ارسال پاسخ به کاربر را شامل می شود.

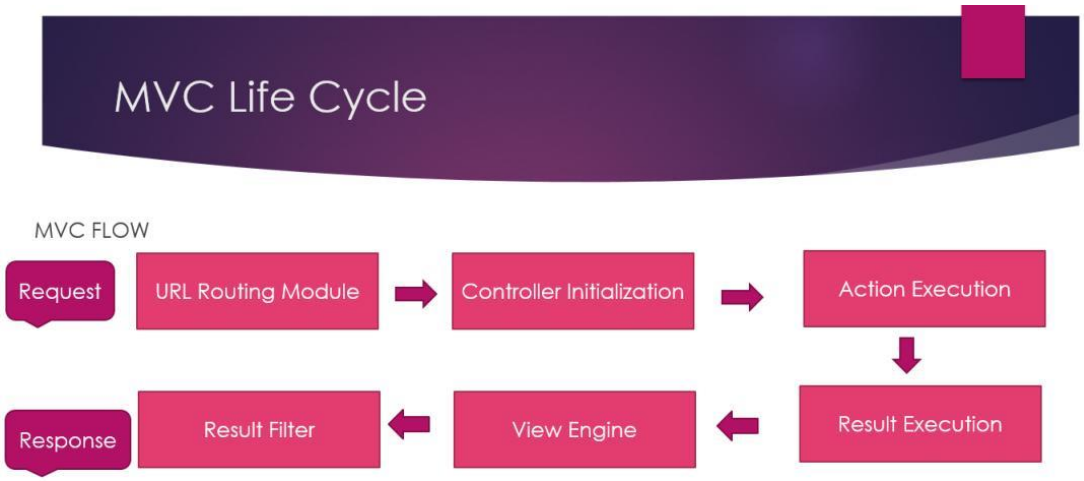
اگر شما یک برنامه نویس mvcهستید حتما میدانید که این طراحی به شما قدرت زیادی در توسعه نرم افزار می دهد در ضمن سرعت تولید پروژه های شما را تا حد زیادی گسترش می دهد .درست است که Asp.net Web Formهمچنان به کار خود ادامه می دهد ولی mvcجایگزین کامل و قدرتمندی برای آن است .همان طور که می دانید mvcاز تکنیک Agile(برای آشنایی با این مفهوم به مقاله آشنایی با متدولوژی Agile برای تیم های توسعه دات نت مراجعه کنید .)هم استفاده می کند .هدف از Agileارائه نرم افزارهایی کارا و با کیفیت به مشتری است.

کار با Aspمشکلات خاص خود را داشت . کار با view Stateها حجم درخواست های ما را افزایش می داد View State. مکانیزمی برای نگه داری وضعیت صفحه در حین رد و بدل شدن صفحات در حین جابه جایی بین سرور و کلاینت است view state. موجب از بین رفتن پهنای باند و کاهش سرعت ما می شد.مشکل دیگری که وجود داشت این بود که چرخه حیات صفحه بسیار طولانی و پیچیده بود ولی این پیچیدگی در mvcبسیار کمتر شده است.

شاید به نظر برسد که در aspبه خاطر وجود code behindلایه های نمایش و منطق را از هم جدا کرده ایم ولی در واقع این طور نیست.

کنترلی که بر روی خروجی دارید بسیار کم است .و این باعث می شود با جاوااسکرپت به سختی بتوان تغییرات را صورت داد.

چرخه حیات برنامه ها در MVCدر زیر نشان داده شده است



نقطه شروع برنامه های mvcدر واقع از routingشروع می شود .وقتی که پلت فرم ASP.NETدرخواستی را دریافت می کند تصمیم می گیرد که بر اساس URLآن را چگونه مدیریت کند .ماژول Routingدرخواست را دریافت می کند .اگر این درخواست با الگوهای آدرسی که در قسمت Rout.configخود تعریف کرده ایم مطابقت داشت با سمت آن کنترلر و اکشن خاص هدایت می شود .دو کلاسی که در اینجا به کار مدیریت کردن درخواست های ورودی می پردازند کلاس های MVC RouteHandlerو MVC HTTPHandlerهستند .این دو کلاس به عنوان دروازه های mvcمی باشند.

بعد از این که درخواست با کنترلر و اکشن خاصی mapشد نوبت به پردازش کنترلر می رسد .در این مرحله تزریق وابستگی هم می تواند اتفاق بیافتد .که الگوی تزریق در اینجا factoryاست .وقتی درخواستی می رسد factoryپیش فرض یا همان DefaultControllerFactoryبه بررسی route dataمی پردازد تا کنترلر را بیابد .و بعد به دنبال کلاس کنترلر می گردد .بعد از این Controller Activatorیک نمونه از کلاس کنترلر یافته شده ایجاد می کند .مرحله بعد اجرای اکشن مورد نظر می باشد .

کاری که انجام می دهد این است که اکشن مورد نظر را برای اجرا پیدا کند .اگر خروجی اکشن از نوع viewResult بود در ادامه view engine هم فراخوانی می شود .تا خروجی مورد نظر را نمایش دهد. action invoker

حال شروع به کار عملی می کنیم .یک پروژه خالی mvc ایجاد کنید .نکته مهمی که باید آن را مد نظر داشته باشید این است که فایل global.asax که در داخل آن کلاس MvcApplication وجود دارد بسیار مهم است . و نقطه شروع برنامه می باشد .این کلاس از HttpApplication مشتق می شود .این کلاس چند رویداد مهم را که باعث اجرای برنامه ما می شود را شروع می کند

ApplicationStart

BeginRequest

ResolveRequestCache

MapRequestHandler

AcquireRequestState

RequestHandlerExecute

UpdateRequestCache

LogRequest

EndRequest

ApplicationEnd

بنابر این می توان گفت هر برنامه mvc ابتدا از Application_Start شروع می شود .و این متد ابتدا در داخل خود تمام مسیر های تعریف شده در سیستم را فراخوانی کرده است .این متد تمام این مسیرها را به Route Table اضافه می کند .حال یک کالکشنی داریم که در آن مسیرهای مورد قبول ما ثبت شده اند و می توان آنها را به RouteHandler برای مسیریابی تحویل داد . نقطه مقابل متد Application_Start است که برنامه را خاتمه داده و همه منابع را آزاد می کند. Application_End

```
public class MvcApplication : System.Web.HttpApplication
{
   protected void Application_Start()
   {
      AreaRegistration.RegisterAllAreas();
      RouteConfig.RegisterRoutes(RouteTable.Routes);
```

```
        // RouteDebug.RouteDebugger.RewriteRoutesForTesting(RouteTable.Routes);

    }
    protected void Application_BeginRequest(object sender, EventArgs e)
    {
        var persianCulture = new PersianCulture();
        Thread.CurrentThread.CurrentCulture = persianCulture;
        Thread.CurrentThread.CurrentUICulture = persianCulture;
    }
}
```

بعد از application_Start **اول رویداد** BeginRequest **رخ می دهد .اگر در داخل برنامه سیستم مجوز دهی داشته باشیم** AuthorizeRequest **رویداد بعدی است که رخ می دهد .بعد از این رویداد** ResolveRequestCache **رخ خواهد داد .این رویداد اگر** cache **در سیستم استفاده کرده باشیم و بخواهیم به حافظه** cache **دسترسی داشته باشیم اتفاق خواهد افتاد.**

MapRequestHandler **وقتی پلت فرم** asp **قصد انتخاب هندلری را برای پاسخگویی به درخواست رسیده دارد این رویداد اتفاق می افتد**

AcquireRequestState **این رویداد داده هایی همچون** session **را برای ما بازیابی می کند.**

UpdateRequestCache **این رویداد ماژول های** cache **را جهت به روزرسانی محتویات** cache **به روزرسانی می کند.**

LogRequest **قبل از انجام لاگین رخ می‌دهد**

EndRequest **وقتی درخواست تمام شد اجرا می شود.**

در شکل زیر دو متد مهم application_start **و** Application_end **را می بینید .همان طور که شاهد هستید کار خاصی در این کلاس ها علاوه بر آنچه در حالت عادی هستند انجام نداده ایم.**

```
0 references
public class MvcApplication : System.Web.HttpApplication
{
    0 references
    protected void Application_Start()
    {
        AreaRegistration.RegisterAllAreas();
        RouteConfig.RegisterRoutes(RouteTable.Routes);
    }
    0 references
    protected void application_End()
    {
        Debug.WriteLine("Application stopped");
    }
}
```


کلا دو چرخه حیات مهم در mvcداریم که یکی Applicationو دیگری Requestاست .چرخه حیات applicationزمان شروع اجرای برنامه در سرور تا آخر است و چرخه حیات درخواست در واقع از زمان شروع یک درخواست تا ارسال پاسخ به کاربر را شامل می شود.

پنجره output را در شکل زیر مشاهده می کنید که در آن خروج کاربر با کد صفر را بیان کرده است .این خط نشان دهنده اجرای Application_Endمی باشد.

```
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/102/ROOT-1-130999195452596553): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Web.
'iisexpress.exe' (CLR v4.0.30319: DefaultDomain): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Web.RegularExpressions\v4.0_4.
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/102/ROOT-1-130999195452596553): Loaded 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\Temporary
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/102/ROOT-1-130999195452596553): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\Microsoft.CS
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/102/ROOT-1-130999195452596553): Loaded 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\Temporary
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/102/ROOT-1-130999195452596553): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Dynar
A first chance exception of type 'Microsoft.CSharp.RuntimeBinder.RuntimeBinderException' occurred in Microsoft.CSharp.dll
A first chance exception of type 'Microsoft.CSharp.RuntimeBinder.RuntimeBinderException' occurred in Microsoft.CSharp.dll
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/102/ROOT-1-130999195452596553): Loaded 'C:\Windows\Microsoft.NET\Framework\v4.0.30319\Temporary
A first chance exception of type 'Microsoft.CSharp.RuntimeBinder.RuntimeBinderException' occurred in Microsoft.CSharp.dll
A first chance exception of type 'Microsoft.CSharp.RuntimeBinder.RuntimeBinderException' occurred in Microsoft.CSharp.dll
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/102/ROOT-1-130999195452596553): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Runt
'iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/102/ROOT-1-130999195452596553): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\Microsoft.V
The program '[3848] iisexpress.exe: Program Trace' has exited with code 0 (0x0).
The program '[3848] iisexpress.exe' has exited with code 0 (0x0).
```