



# MVC med Javascript och Ajax

Filip Ekberg

# Filip Ekberg

Microsoft & Xamarin MVP

[mail@filipekberg.se](mailto:mail@filipekberg.se)

 smorgasbord

pluralsight 

 fekberg





# Vad ska vi lära oss?

I denna kurs lär sig den studerande hur man jobbar med den naturliga uppdelningen av **data, GUI och logik** i sina projekt.

Detta görs med **MVC** som är ett ramverk. MVC har mer eller mindre blivit en **standard** för avancerade .NET-webbsidor. Arbetssättet gör att koden blir mer **återanvändbar och kostnadseffektiv**.

Man delar upp projektet i de **logiska lagren** i **Modell, View och Controller**. Varje logiskt lager kan ersättas av en ny modul för att skapa en helt ny applikation.

# Vad ska vi lära oss?

- Skapa projekt i ramverket ASP.NET MVC
- Projektstruktur i Visual Studio
- Model-View-Controller:
  - Lagerstruktur och concept
  - Designmönster & Arkitektur
- Integrering av databas och Entity Framework
- Lägga till och modifiera: Layout, Controllers, Razor Views
- JavaScript, Ajax
- Säkerhet & Infrastruktur

# Mål

Målet med kursen är att studerande ska kunna skapa **återanvändbar** kod enligt **MVC-ramverket**. På detta sätt kan studerande skapa webbprojekt som kan växa och expandera på ett kontrollerat sätt.

# Mål

Efter genomgången kurs ska den studerande ha kunskaper och färdigheter inom

- Ramverket ASP.NET MVC med dess komponenter och tekniker.
- JavaScript och Ajax för dynamiska responsiva hemsidor
- Integrering av MVC ramverket och Entity Framework ramverket.

# För att få betyget **godkänt**

- Har erhållit lägst betyget godkänt på tentamen
- Har klarat av alla obligatoriska moment
- Kan förklara MVC-modellens lagerstruktur
- Ska kunna skapa MVC-applikationer enligt god programmeringssed
- Kan lägga till och modifiera: Layout, Controllers, Razor Views
- Kan skapa responsiva hemsidor i MVC
- Förstår hur JavaScript och ASP.NET MVC arbetar tillsammans
- Förstår kopplingen mellan DOM (Document Object Model) och Ajax
- Kan skriva enklare JavaScript som används i MVC-projekt

# För att få betyget **väl godkänt**

- Uppnått kunskapskraven för betyget godkänt
- Har erhållit lägst betyget **väl godkänt** på tentamen
- Kan använda MVC-projekt obehindrat
- Kan göra korrekta ändringar i modellen obehindrat
- Kan använda DOM-modellen med Ajax-kod
- Har fördjupad förståelse för JavaScript



# Exemination

Redovisning av obligatoriska moment  
(muntligt/skriftligt)

Individuell skriftlig tentamen

THE EXPERT'S VOICE® IN ASP.NET

FIFTH EDITION

# Pro ASP.NET MVC 5

*BUILD THE MOST MAINTAINABLE, STANDARDS-  
COMPLIANT, AND BEST PERFORMING WEB  
APPLICATIONS ON THE MICROSOFT PLATFORM*

Adam Freeman

Apress®



PLURALSIGHT

# Upplägg

8.30 - 10.00 (1.5h Föreläsning)

10.20 - 12.00 (1.5h Föreläsning)

13.00 - 16.30 (Labb och Självstudie)



# Upplägg

## Vecka 1

Lektion 1 - Introduktion

Lektion 2 - Introduktion

Fortsätter

## Vecka 3

Lektion 5 - Datalagring

Lektion 6 - Testning

Lektion 7 - Säkerhet & Avancerad MVC

## Vecka 2

Lektion 3 - Javascript & Ajax

Lektion 4 - Javascript & Ajax

Fortsätter

## Vecka 4

Lektion 8 - Säkerhet fortsätter

Lektion 9 – Repetition, förberedelser för tentamen







# MVC med Javascript och Ajax

Lektion 1 - Introduktion

# Dagens mål

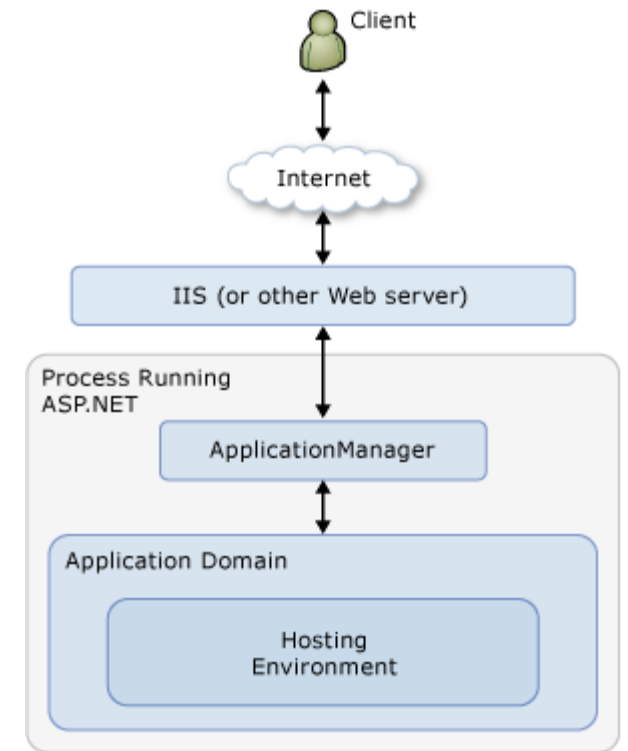
- Vad är MVC? Hur skiljer det sig från vad vi gjort tidigare?
- Djupdykning i MVC som ett mönster
- Hur skapar vi ett nytt projekt som bygger på MVC - Tomt vs. "Template" - Vilka referenser behövs
- Vad innerhåller projektet?
- Hur fungerar request-pipelinene?
- "Startup" och Middlewares

# Förkunskapskrav



# IIS

- Vad gör en webbserver?
- Varför IIS?
- Vad är ASP.NET?





# C# och Webben

- ASP.NET
- ASP.NET MVC
- Varför C#?
- Dela kod mellan desktop och webb

**127.0.0.1**

**fekberg.com**

# **XML, JSON, HTML**

Content Types

# Stateless vs Stateful



# REST

Representational state transfer

# HTTP Verbs

GET, POST, PUT, DELETE, ...

# MVC

a **software design pattern** is a  
general reusable solution to a  
commonly occurring problem within  
a given context in software design

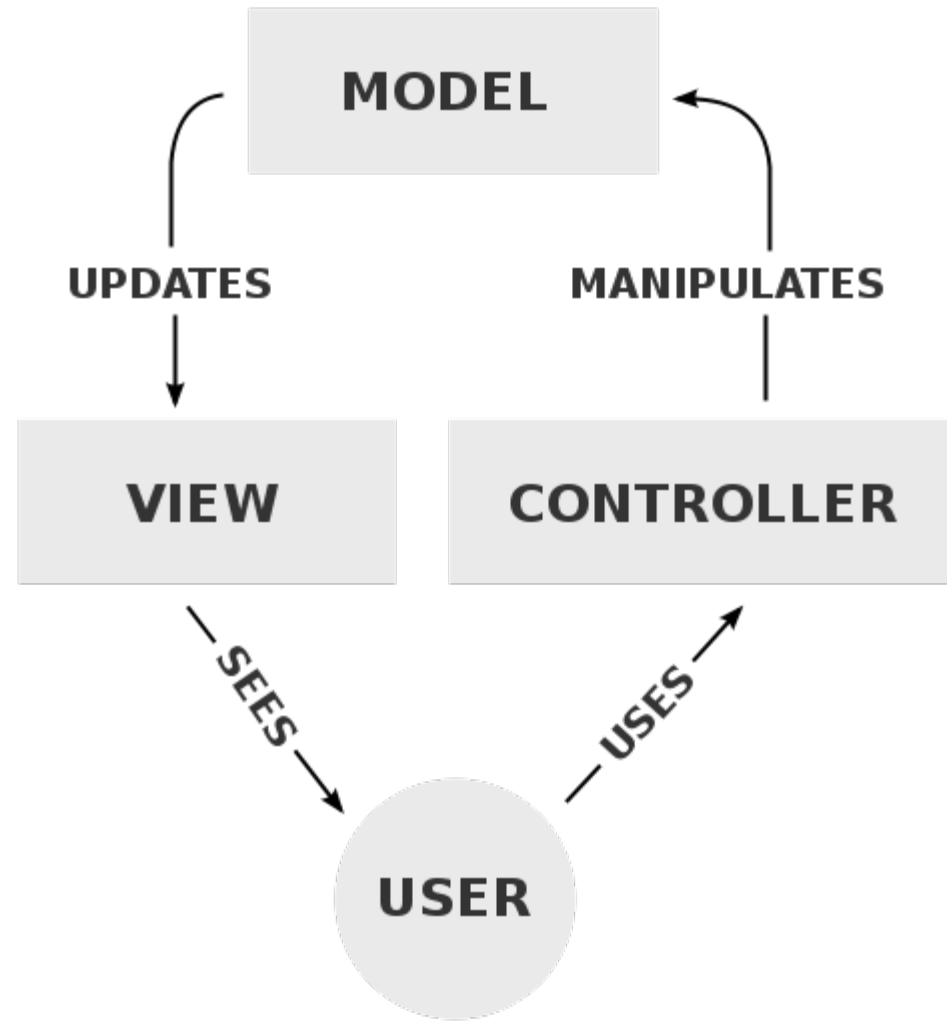
# Varför MVC?

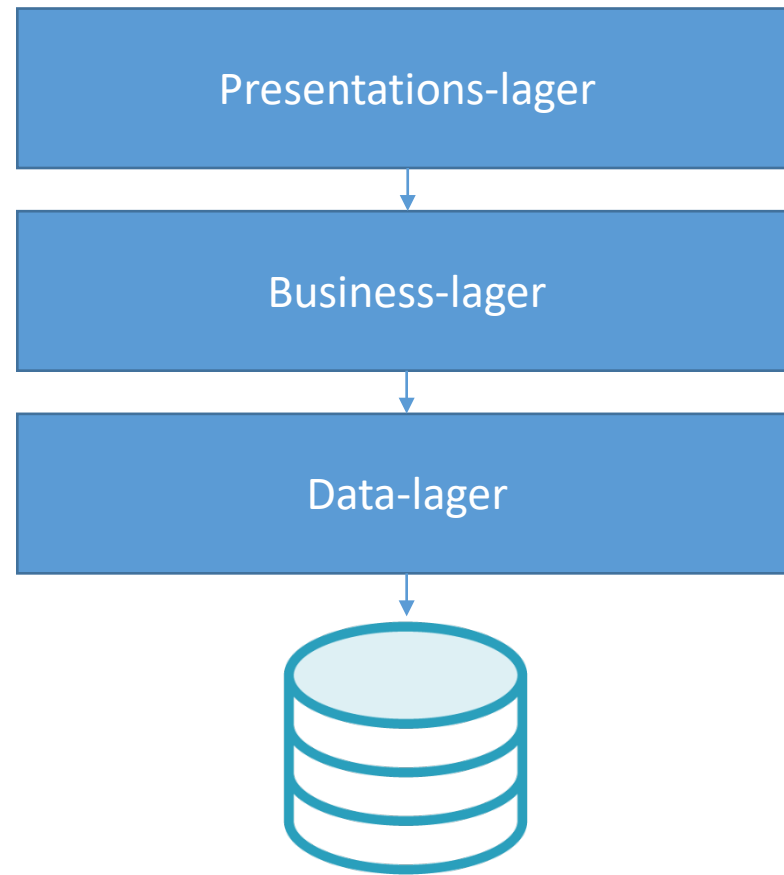
**Uppdelning** av lager

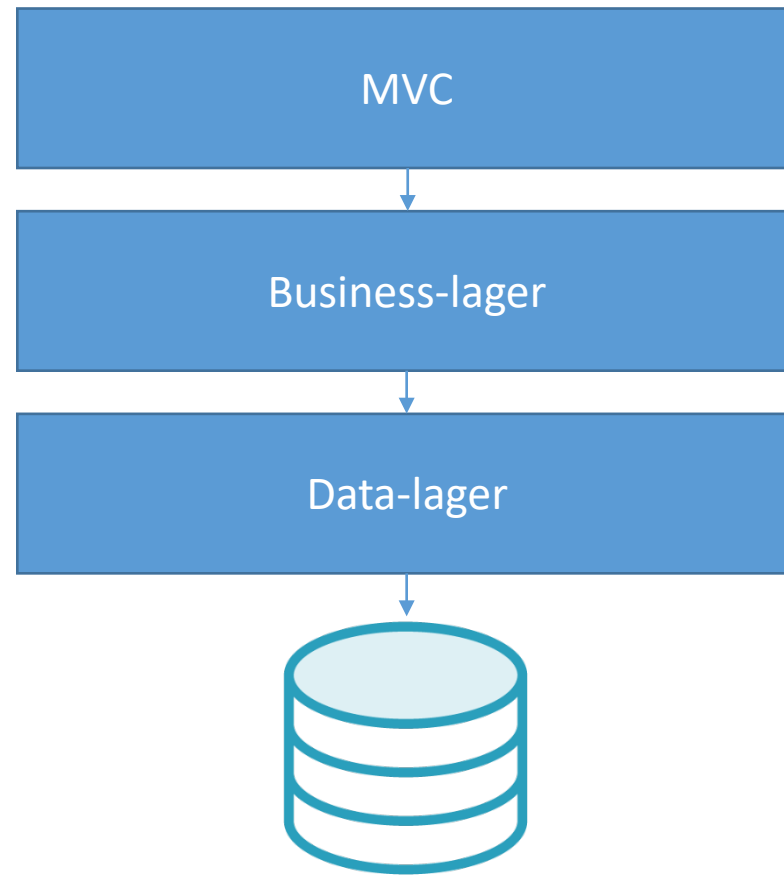
**Testbarhet**

Anpassat för **webb** (stateless)









# Release history

Date	Version
10 December 2007	ASP.NET MVC <a href="#">CTP</a>
13 March 2009	ASP.NET MVC 1.0 <sup>[10]</sup>
16 December 2009	ASP.NET MVC 2 <a href="#">RC</a> <sup>[11]</sup>
4 February 2010	ASP.NET MVC 2 RC 2 <sup>[12]</sup>
10 March 2010	ASP.NET MVC 2 <sup>[13]</sup>
6 October 2010	ASP.NET MVC 3 Beta <sup>[14]</sup>
9 November 2010	ASP.NET MVC 3 RC <sup>[14]</sup>
10 December 2010	ASP.NET MVC 3 RC 2 <sup>[15]</sup>
13 January 2011	ASP.NET MVC 3 <sup>[16]</sup>
20 September 2011	ASP.NET MVC 4 Developer Preview <sup>[17]</sup>
15 February 2012	ASP.NET MVC 4 Beta <sup>[18]</sup>
31 May 2012	ASP.NET MVC 4 RC <sup>[19]</sup>
15 August 2012	ASP.NET MVC 4 <sup>[20]</sup>
30 May 2013	ASP.NET MVC 4 4.0.30506.0 <sup>[21]</sup>
26 June 2013	ASP.NET MVC 5 Preview <sup>[22]</sup>
23 August 2013	ASP.NET MVC 5 RC 1 <sup>[23]</sup>
17 October 2013	ASP.NET MVC 5 <sup>[23]</sup>
17 January 2014	ASP.NET MVC 5.1 <sup>[23]</sup>
10 February 2014	ASP.NET MVC 5.1.1 <sup>[23]</sup>
4 April 2014	ASP.NET MVC 5.1.2 <sup>[23]</sup>
22 June 2014	ASP.NET MVC 5.1.3 <sup>[23]</sup>
1 July 2014	ASP.NET MVC 5.2.0 <sup>[23]</sup>
28 August 2014	ASP.NET MVC 5.2.2 <sup>[23]</sup>
9 February 2015	ASP.NET MVC 5.2.3 <sup>[23]</sup>
6 November 2014	ASP.NET MVC 6.0.0-beta1 <sup>[24]</sup>
18 November 2015	ASP.NET MVC 6.0.0-rc1 <sup>[24]</sup>
17 May 2016	ASP.NET Core MVC 1.0.0-rc2 <sup>[24]</sup>
12 August 2016	ASP.NET Core MVC 1.0.0 <sup>[24]</sup>
17 August 2016	ASP.NET Core MVC 1.0.1 <sup>[24]</sup>

# Model

# C# Klass

Ett "kontrakt" utan logik

```
public class Product
{
    public Guid Id { get; set; }
    public string Name { get; set; }
    public decimal Price { get; set; }
    public decimal Tax { get; set; }
}
```

# View





# Razor

View Engine – Hur ska modellen representeras?

@model Product

```
<h1>@Model.Name</h1>
```

```
<div>
```

```
    <h2>Price</h2>
```

```
    <span>@Model.Price</span>
```

```
    <h2>VAT</h2>
```

```
    <span>@Model.Tax</span>
```

```
    <a href="/cart/add/@Model.Id">Purchase</a>
```

```
</div>
```

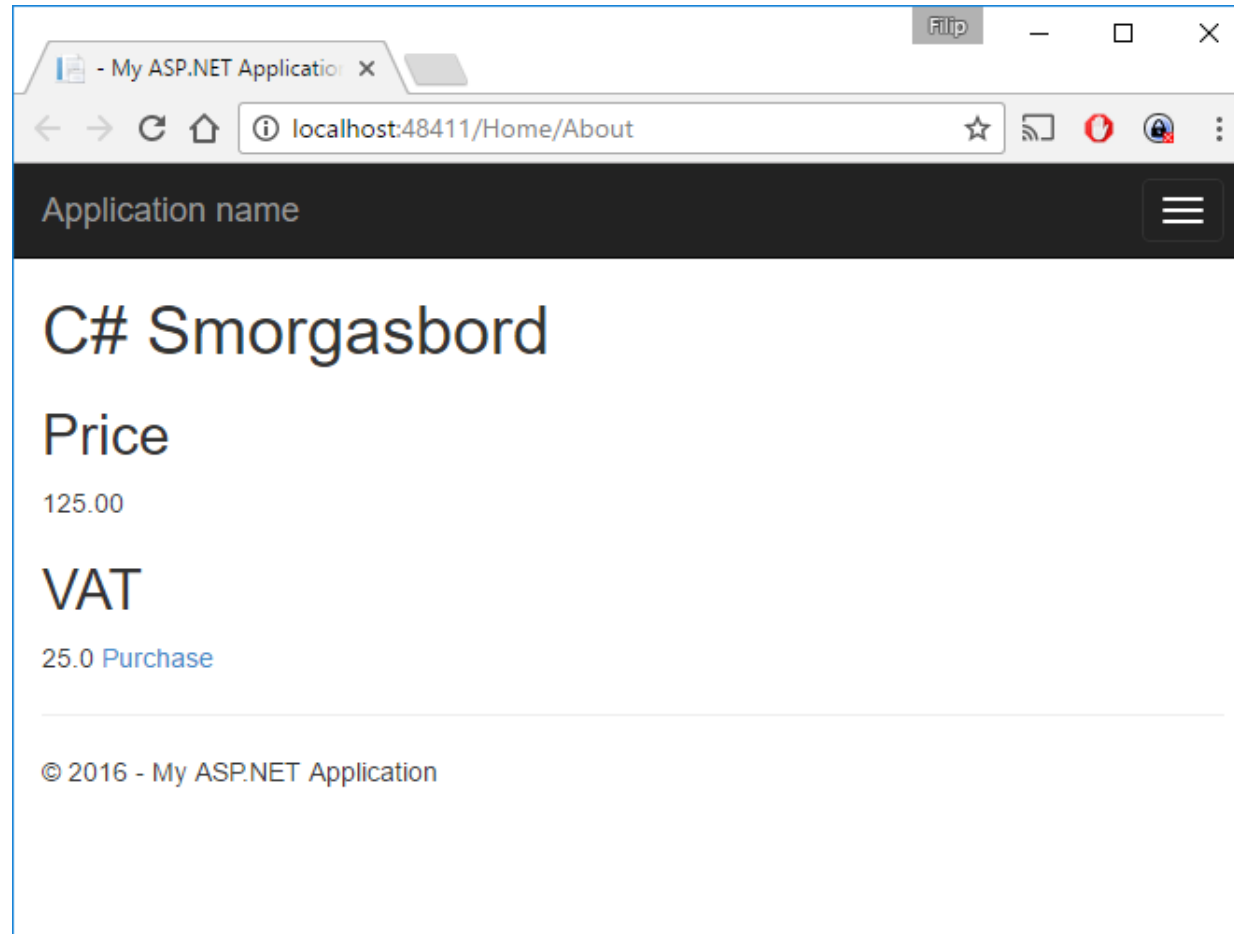
# Controller & Action

# Vem tar hand om vad?

```
public class HomeController : Controller
{
}
}
```

```
public ActionResult Product(Guid id)
{
    var model = new Product
    {
        Id = Guid.NewGuid(),
        Name = "C# Smorgasbord",
        Price = 125.00m,
        Tax = 25.0m
    };

    return View(model);
}
```



# Microservices

Små & Enkla Controllers/Actions



The background of the image is a dark, blurred screenshot of a code editor. It shows various lines of code in different colors (blue, green, yellow, and white) on a dark background. Some legible fragments include 'thumbnails\_columns', 'attachment\_url( \$attachment\_id );', and '\$attachment\_id, \$file\_name'.

# Hur funkar det?

/Home/About



/ {Controller} / {Action}



```
public class HomeController : Controller
{
    public ActionResult About()
    {
        return Content("Alright!");
    }
}
```

/Gallery/1/Sunrise-at-the-Beach

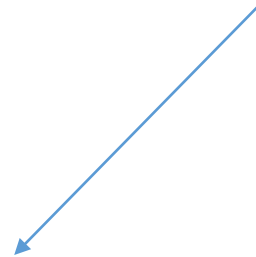


/ {Controller} / {Id} / {Meta}

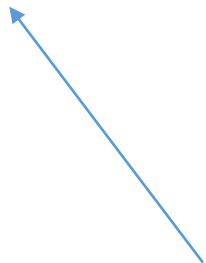


```
public class GalleryController : Controller
{
    public ActionResult Show(int id, string meta)
    {
        return Content("OK!");
    }
}
```

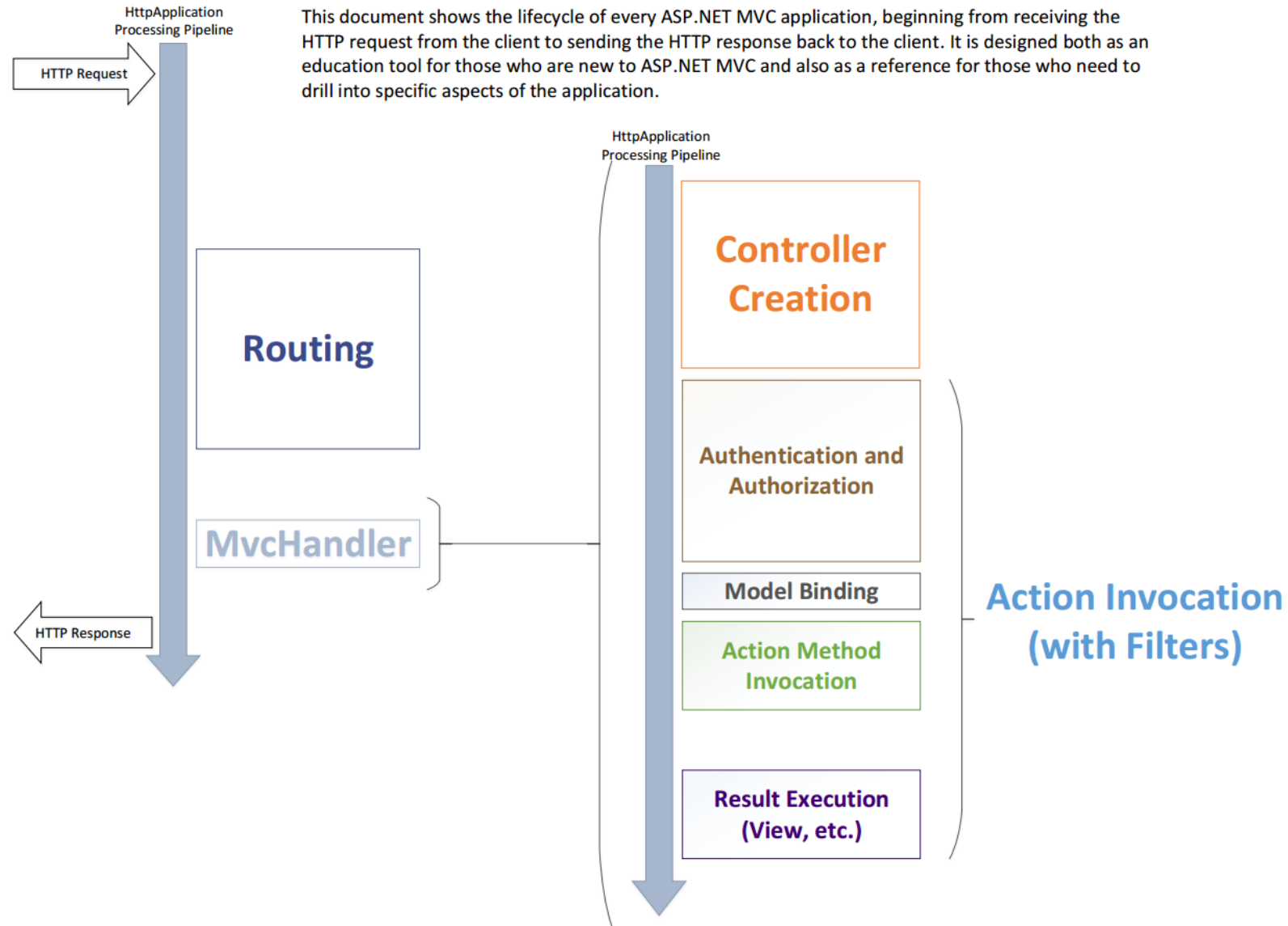
Sunrise-at-the-Beach



1



# ASP.NET MVC 5 APPLICATION LIFECYCLE – HIGH-LEVEL VIEW



```
public class RouteConfig
{
    public static void RegisterRoutes(RouteCollection routes)
    {
        routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

        routes.MapRoute(
            name: "GalleryRoute",
            url: "Gallery/{id}/{meta}",
            defaults: new { controller = "Gallery", action = "Show",
                           id = UrlParameter.Optional,
                           meta = UrlParameter.Optional }
        );

        routes.MapRoute(
            name: "Default",
            url: "{controller}/{action}/{id}",
            defaults: new { controller = "Home",
                           action = "Index",
                           id = UrlParameter.Optional }
        );
    }
}
```

```
public string Data { get; set; }  
public ActionResult Show(int id, string meta)  
{  
    if(string.IsNullOrEmpty(Data)) Data = meta;  
  
    return Content(Data);  
}
```



/Gallery/1/Sunrise-at-the-Beach

Sunrise-at-the-Beach



/Gallery/2/


? Inget!

**Varför sparas inte  
värdet i min Property?**



# Stateless!





```
public static string Data { get; set; }  
public ActionResult Show(int id, string meta)  
{  
    if(string.IsNullOrEmpty(Data)) Data = meta;  
  
    return Content(Data);  
}
```



/Gallery/1/Sunrise-at-the-Beach

Sunrise-at-the-Beach



/Gallery/2/

Sunrise-at-the-Beach

# Global.asax

```
public class MvcApplication : System.Web.HttpApplication
{
    protected void Application_Start()
    {}

    protected void Application_End()
    {}

    protected void Session_Start()
    {}

    protected void Session_End()
    {}

    protected void Application_Error()
    {}
}
```

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

# Web.config

# Web.config

- Inställningar - App Settings
- Databasinformat
- Vilken .NET version
- HTTP Handlers
- URL Rewrite
- Ändrar du i Web

```
<appSettings>
  <add key="webpages:Version" value="3.0.0.0"/>
  <add key="webpages:Enabled" value="false"/>
  <add key="ClientValidationEnabled" value="true"/>
  <add key="UnobtrusiveJavaScriptEnabled" value="true"/>
  <add key="MySetting" value="SomeValue" />
</appSettings>
<system.web>
  <compilation debug="true" targetFramework="4.5.2"/>
  <httpRuntime targetFramework="4.5.2"/>
</system.web>
```

# Mer Web.config

- Web.config för Views-katalogstrukturen
- Inkludera default namespace i dina vyer

```
<namespaces>  
  <add namespace="System.Web.Mvc" />  
  <add namespace="System.Web.Mvc.Ajax" />  
  <add namespace="System.Web.Mvc.Html" />  
  <add namespace="System.Web.Routing" />  
  <add namespace="MyMvcApplication" />  
</namespaces>
```

# System.Web vs OWIN



# System.Web

- Tar hand om all hantering av request/respons
- Hostas via IIS eller IIS Express

# OWIN

- Specifikation för hur webbservrar och applikationer kan skrivas löst kopplade
- Tillåter oss att enklare introducera "Middlewares"

```
[assembly: OwinStartup(typeof(MyMvcApp.App_Start.Startup))]  
  
namespace MyMvcApp.App_Start  
{  
    public class Startup  
    {  
        public void Configuration(IAppBuilder app)  
        {  
            AreaRegistration.RegisterAllAreas();  
  
            RouteConfig.RegisterRoutes(RouteTable.Routes);  
  
            app.UseCookieAuthentication(new CookieAuthenticationOptions  
            {  
                AuthenticationType = "Cookie",  
                AuthenticationMode = AuthenticationMode.Active,  
                LoginPath = new PathString("/login"),  
                LogoutPath = new PathString("/logout"),  
            });  
        }  
    }  
}
```

Install-Package Microsoft.Owin.Host.SystemWeb

# Lär känna ASP.NET MVC

# Lär känna ASP.NET MVC

- Sätt breakpoints överallt
- Ta bort, Lägg Till och Ändra för att se hur det påverkar
- IIS Express kan ibland behövas tvingas stängas av

# Labb

# Bildgalleri



# Bildgalleri?

- Ladda upp bilder
- Skapa album av uppladdade bilder
- Kommentera på bilder
- Inloggning & Säkerhet
- Administrationsgränssnitt

