MVC med Javascript och Ajax

Filip Ekberg

Hur går det?

Föreläsningstempo

Går det för fort?

Labbar & Övningar

Är de otydliga?

Behövs mer hjälp?

Upplägg

Mindre föreläsning? Mer labbar?

Läser ni boken?

Läsanvisningar finns på Ping Pong

Hur många timmar per vecka lägger du?

Heltidsstudier!

Stödmaterial

Har ni hjälp av PowerPoint-filerna?
Pluralsight?

MVC med Javascript och Ajax

Lektion 5 - Datalagring

mail@filipekberg.se

fekberg

Hur har det gått?

Vad är svårast att förstå?

Dagens mål

- Repetition
  - Hur skapa man ett tomt MVC-projekt
  - Hur introducerar vi en Controller, Action, Model och View
  - Hur gör vi en GET och POST
  - Hur lägger vi på AJAX
  - Hur manipulerar vi DOM via JavaScript
  - Hur hanterar ASP.NET instancer av objekt (Controller, Modeller)
- Introducera datalagring
  - Utnyttja Session och Cookies
  - Lägga på ett datalager med EntityFramework
  - Förstå skillnaden på datamodell och vymodell
  - Hur hänger det ihop med ASP.NET life cycle
  - Använda ett Repository Pattern

MVC

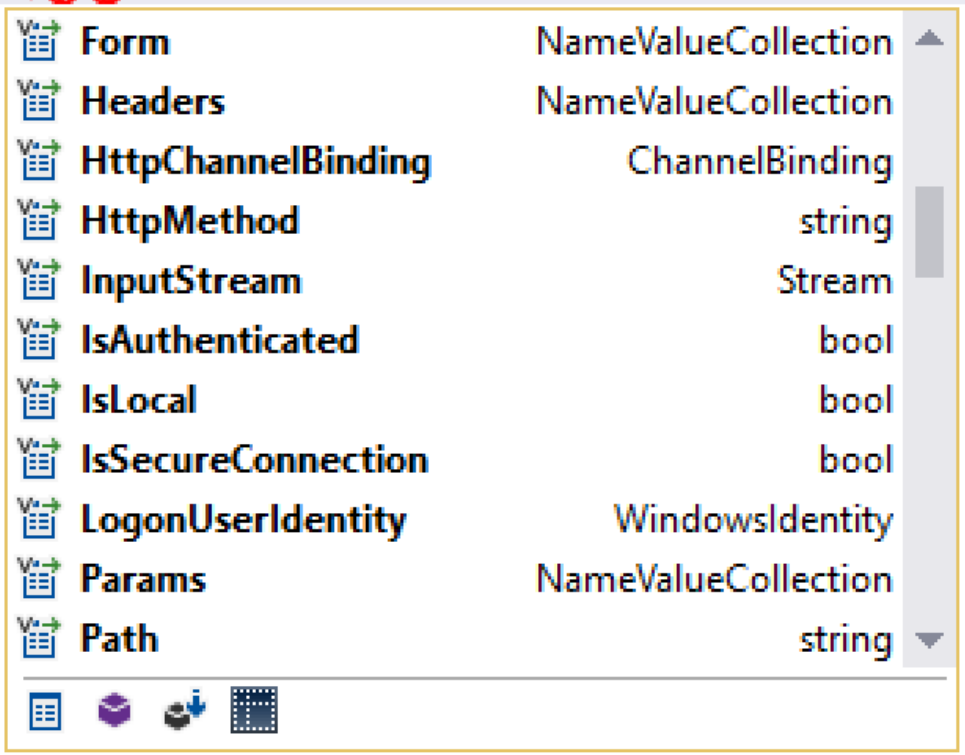
Vilka filer kan jag ladda hem och vilka kompileras av ASP.NET i runtime?

```
1 reference
public class PeopleController : Controller
{
    0 references
    public PeopleController()
    {

    }

    0 references
    public ActionResult Index()
    {
        return View();
    }
}
```

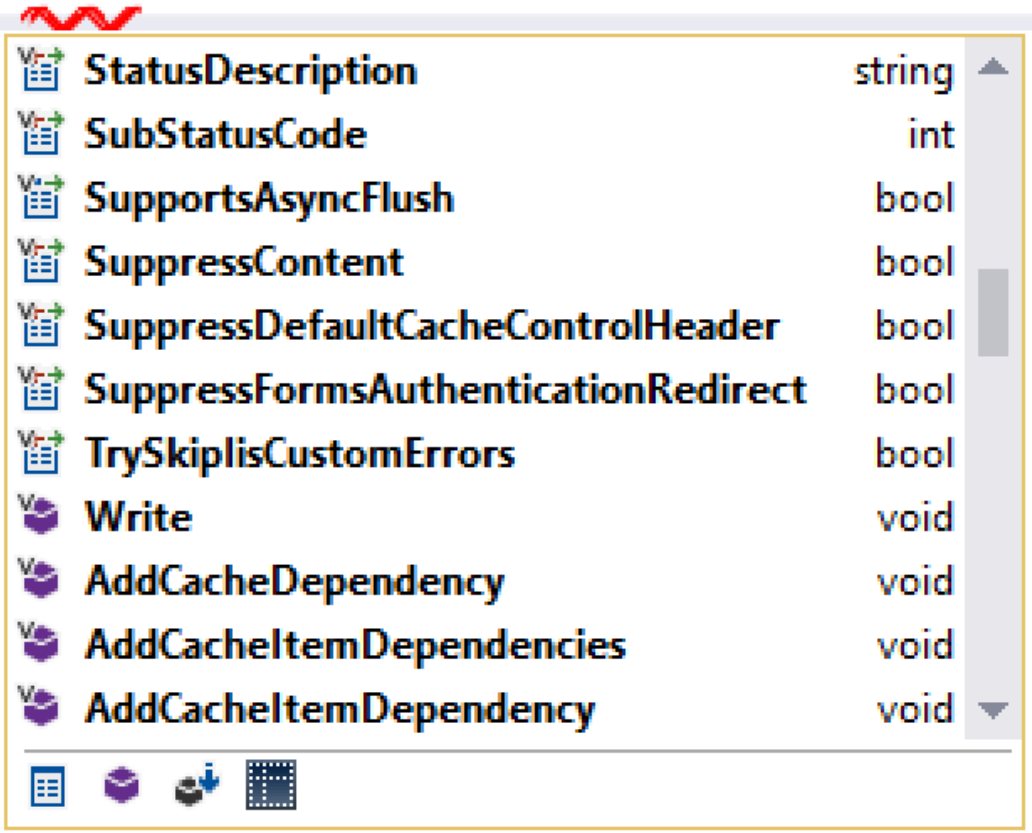
Hur ofta körs konstruktorn av en Controller?

Allt handlar om Request & Response

```
Request.
```


Form NameValueCollection
Headers NameValueCollection
HttpChannelBinding ChannelBinding
HttpMethod string
InputStream Stream
IsAuthenticated bool
IsLocal bool
IsSecureConnection bool
LogonUserIdentity WindowsIdentity
Params NameValueCollection
Path string

Response.

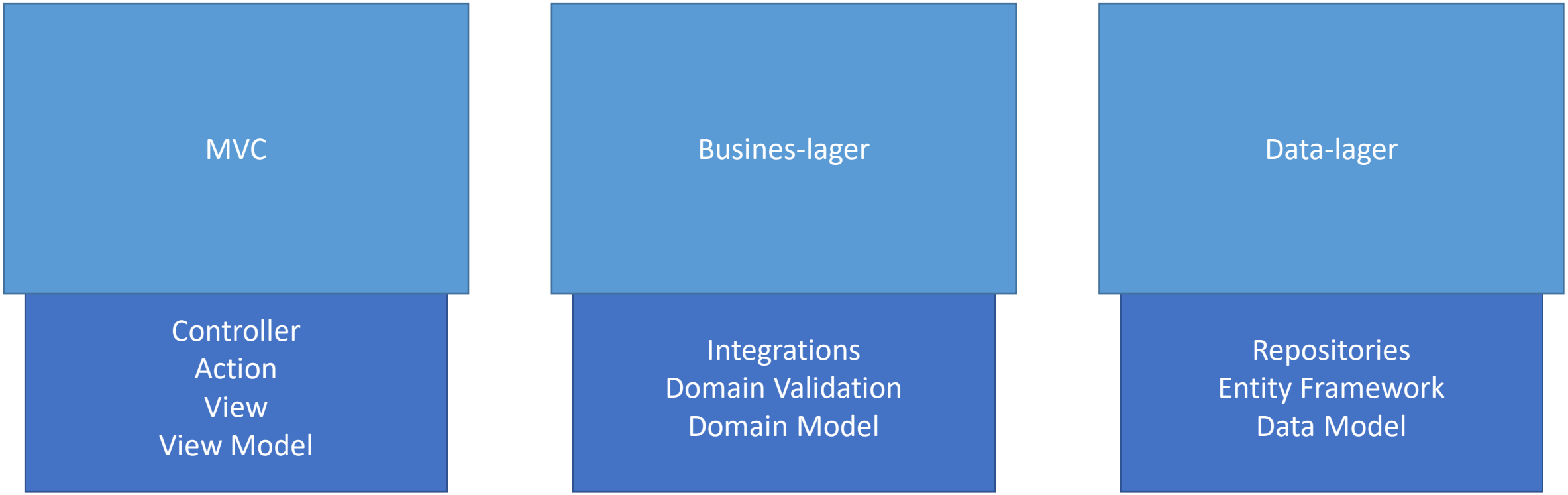

StatusDescription string
SubStatusCode int
SupportsAsyncFlush bool
SuppressContent bool
SuppressDefaultCacheControlHeader bool
SuppressFormsAuthenticationRedirect bool
TrySkiplisCustomErrors bool
Write void
AddCacheDependency void
AddCacheItemDependencies void
AddCacheItemDependency void

Var ska min logik vara?

Undvik logik i vyer!

(Det finns undatag)

”Löst Kopplat”


MVC
Busines-lager
Data-lager
Controller
Action
View
View Model
Integrations
Domain Validation
Domain Model
Repositories
Entity Framework
Data Model

Hur lätt är det att anpassa efter AJAX?

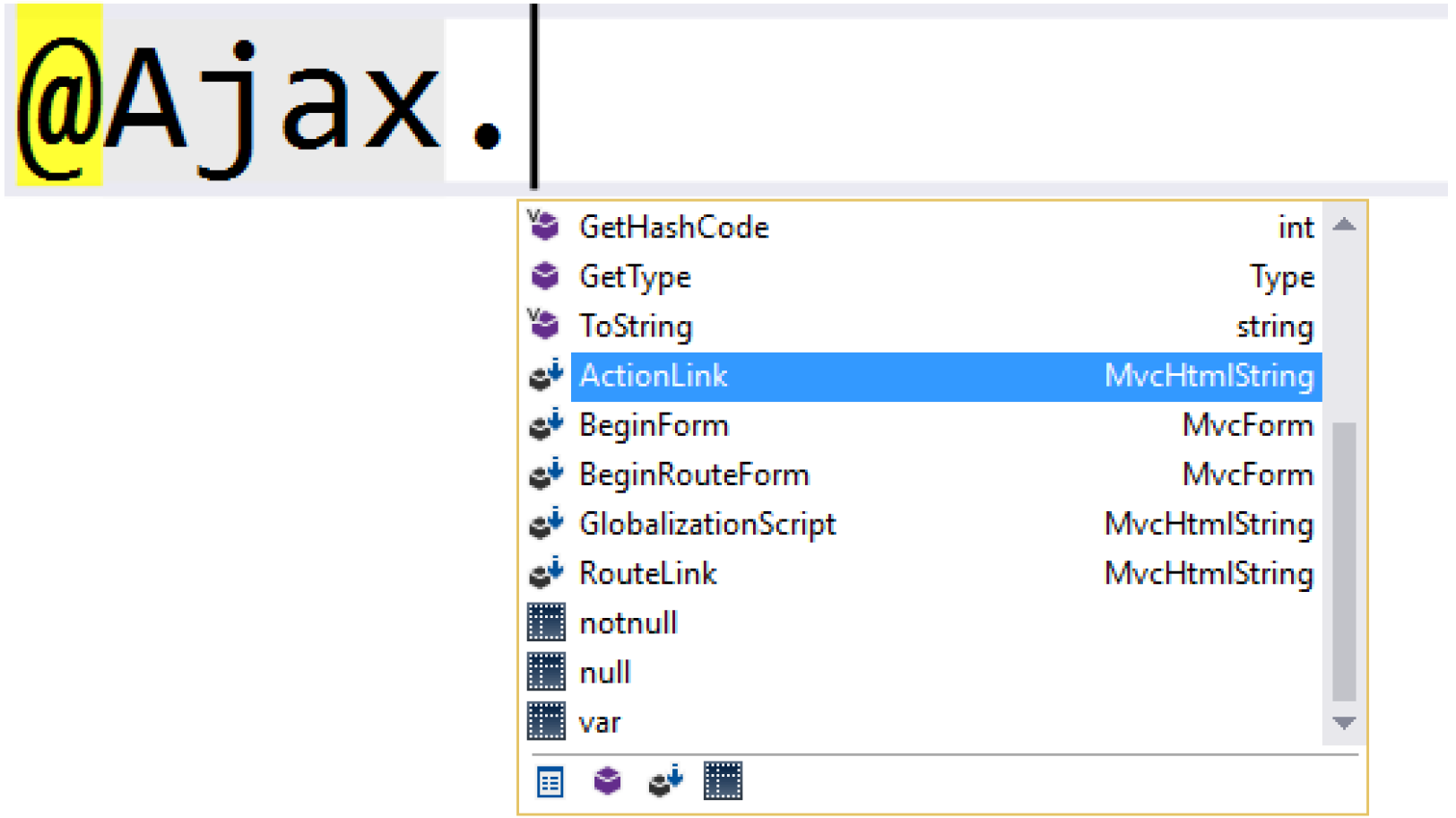
Hur lätt är det att ändra så vi returnerar PartialView istället?

Vad har det för sido-effekter?

AjaxExtensions

Har ni läst boken?

Kapitel 23


@Ajax.
GetHashCode int
GetType Type
ToString string
ActionLink MvcHtmlString
BeginForm MvcForm
BeginRouteForm MvcForm
GlobalizationScript MvcHtmlString
RouteLink MvcHtmlString
notnull
null
var

Istället för att använda Html Extension använder vi nu Ajax Extension

Inställningar för hur AJAX-anropet skall ske

```
@using (Ajax.BeginForm("Create", ajaxOptions))
{
    @Html.TextBoxFor(user => user.Name)

    <input type="submit" value="Lägg till användare" />
}

<div id="result"></div>
```

När AJAX-anropet är klart vill vi uppdatera vår div med Id ”result”

```
@{
    var ajaxOptions = new AjaxOptions
    {
        Confirm = "Är du säker?"
        HttpMethod = "POST",
        InsertionMode = InsertionMode.Replace,
        UpdateTargetId = "result"
    };
}
```

Innehållet i ”result” ersätts

När AJAX-anropet är klart vill vi uppdatera vår div med Id ”result”

```
Install-Package Microsoft.jQuery.Unobtrusive.Ajax
```

_LayoutPage

```
<script src="/Scripts/jquery-3.1.1.min.js"></script>
<script src="/Scripts/jquery.unobtrusive-ajax.min.js"></script>
```

Funkar ej för fil-uppladdning!

Använd jQuery för det!

Underlättar för allt annat – som inte är fil-uppladdning!

Hur vet vi om ett anrop kommer som ett AJAX-anrop?

```
Request.Headers["X-Requested-With"] == "XMLHttpRequest"
```

Gallery/Index Title

localhost:58948/Gallery

Detta kommer från Index.cshtml

Ditt IP: **::1** (Visar ditt IP)
QueryString: (Visar allt efter ? i adressfältet)
Raw URL: **/Gallery** (Visar URLen som du skrivit in, exklusive hostname)

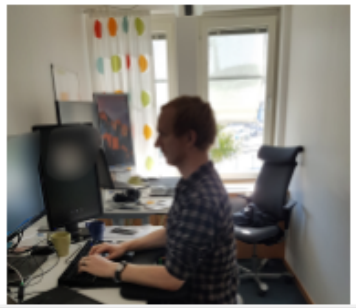
Nu laddar vi in en Partial View!

Ladda upp en ny bild

Test
Choose File 20161004_123655.jpg Ladda upp

Nu tittar vi på en lista av bilder!

Test



Elements Console Sources Network Timeline Profiles Application Security Audits AdBlock

View: Preserve log Disable cache Offline No throttling

Filter Regex Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other

100 ms 200 ms 300 ms 400 ms 500 ms 600 ms 700 ms 800 ms 900 ms 1000 ms

Name

Upload
Upload
List
20161004_123655.jpg

4 requests | 845 KB transferred

Headers Preview Response Cookies Timing

Request Headers view source

Accept: */*
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.8,da;q=0.6,is;q=0.4,nb;q=0.2,sv;q=0.2
Connection: keep-alive
Content-Length: 849745
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryDjGq4Ozo43G1Y0Q9
Cookie: ASP.NET_SessionId=3p4424auexay2v53ho0pzn3l; __RequestVerificationToken=BRN-gAreBLNhEN1Mz46TdF64qm1T1euhsc_KLS9ANLgKrrcyUDDoN9-i8WkpoHtmpSlGvg4Wwh1OOugjwVYLxzYBV1QVzi3ILeP4LATuzHI1; TheCookie=Filip Ekberg
Host: localhost:58948
Origin: http://localhost:58948
Referer: http://localhost:58948/Gallery
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chr
X-Requested-With: XMLHttpRequest

Request Payload

------WebKitFormBoundaryDjGq4Ozo43G1Y0Q9
Content-Disposition: form-data; name="Name"

Console

```
public ActionResult List()
{
    var photos = PhotoRepository.All();

    if (!Request.IsAjaxRequest())
    {
        return View(photos);
    }

    return PartialView(photos);
}
```

Olika vyer beroende på om det är via AJAX eller inte!

Inlämningsuppgift

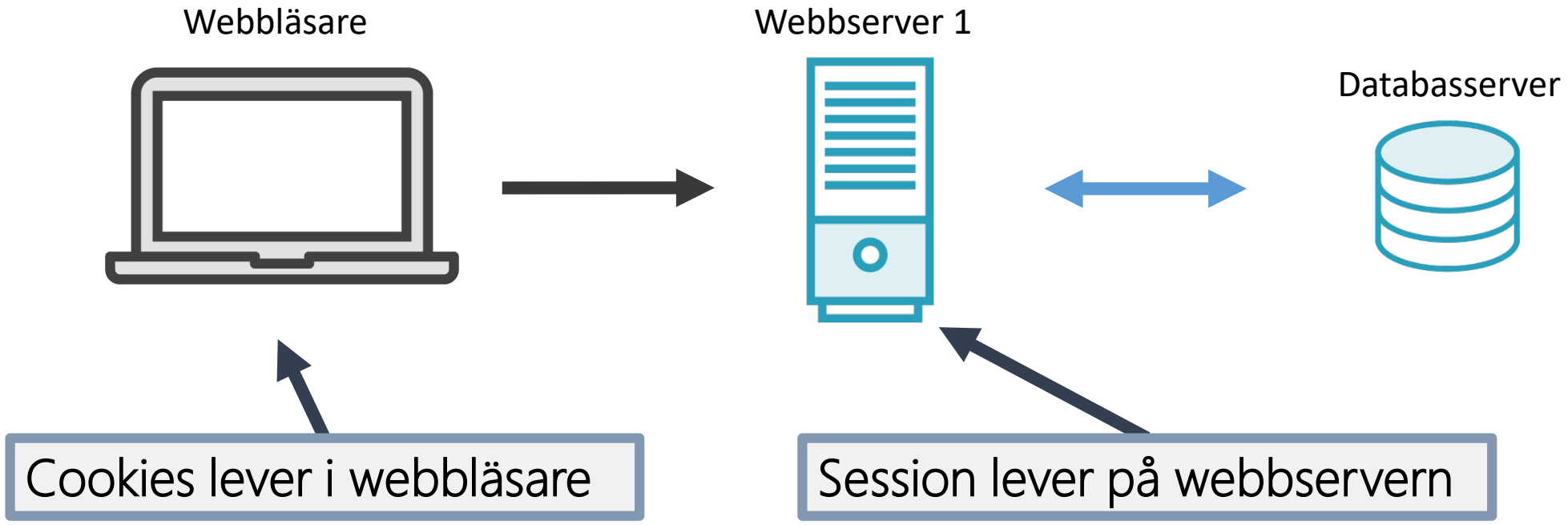
Bygg en adressbok

- Hantering av inlägg i din adressbok
- En Layout som är gemensam för alla dina undersidor
- En Index-sida som ger dig länkar till att lägga till och lista rader i din adressbok
- Adressboken skall minst spara följande: Id, Namn, Telefonnummer, Adress
- Tidpunkt som adressen uppdaterades
- En vy för att lista alla inlägg i adressboken
- En vy för att lägga till ett inlägg i adressboken
- En vy för att ändra/se detaljer om ett inlägg i adressboken
- Lägga Till och Ta bort funktionaliteten skall ske via Ajax
- Listan av adresser skall updateras dynamiskt när du använt Ajax för Lägga till och Ta bort

Lämna in idag senast 17.00

fekberg

Cookies & Session


Webbläsare
Webbserver 1
Databasserver
Cookies lever i webbläsare
Session lever på webbservern

Hur länge lever en Cookie?

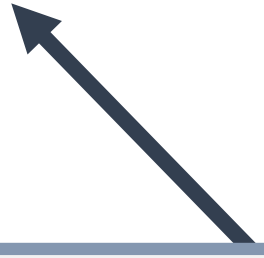
Hur länge lever en Session?

Användaren kan ändra på en Cookie i Webbläsaren

Hur kommer ASP.NET åt Cookies då de kommer från Webbläsaren?

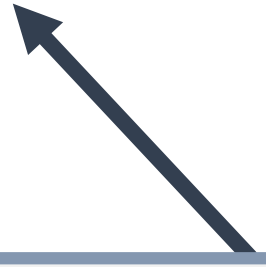
Hur kommer ASP.NET åt Cookies då de kommer från Webbläsaren?

```
Request.Cookies["TheCookie"]
```



Varje request från webbläsaren skickar med alla cookies som stämmer överrens med den domänen man besöker

`Response.Cookies`



Bestäm vilka Cookies som skickas tillbaka till webbläsaren

```
var cookie = new HttpCookie("TheCookie", "Filip Ekberg");
Response.Cookies.Add(cookie);
```

Cookies kan man komma åt via JavaScript!

Om de inte är satta som HttpOnly!

```
var httpOnlyCookie =
     new HttpCookie("TheCookie", "Filip Ekberg")
     {
          HttpOnly = true
     };


var normalCookie = new HttpCookie("SomeCookie",
                                  "ASP.NET MVC Rocks");

Response.Cookies.Add(normalCookie);
Response.Cookies.Add(httpOnlyCookie);
```

Skickas till webbläsaren, men kan inte ändras via JavaScript!

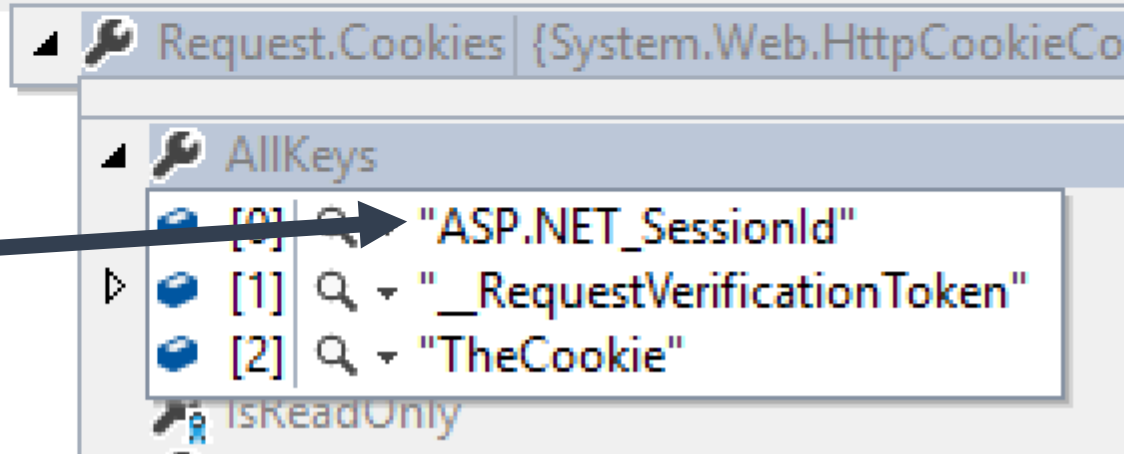
```
> document.cookie
< "SomeCookie=ASP.NET MVC Rocks"
```

En Session kan man inte ändra från webbläsaren – Lever på servern!

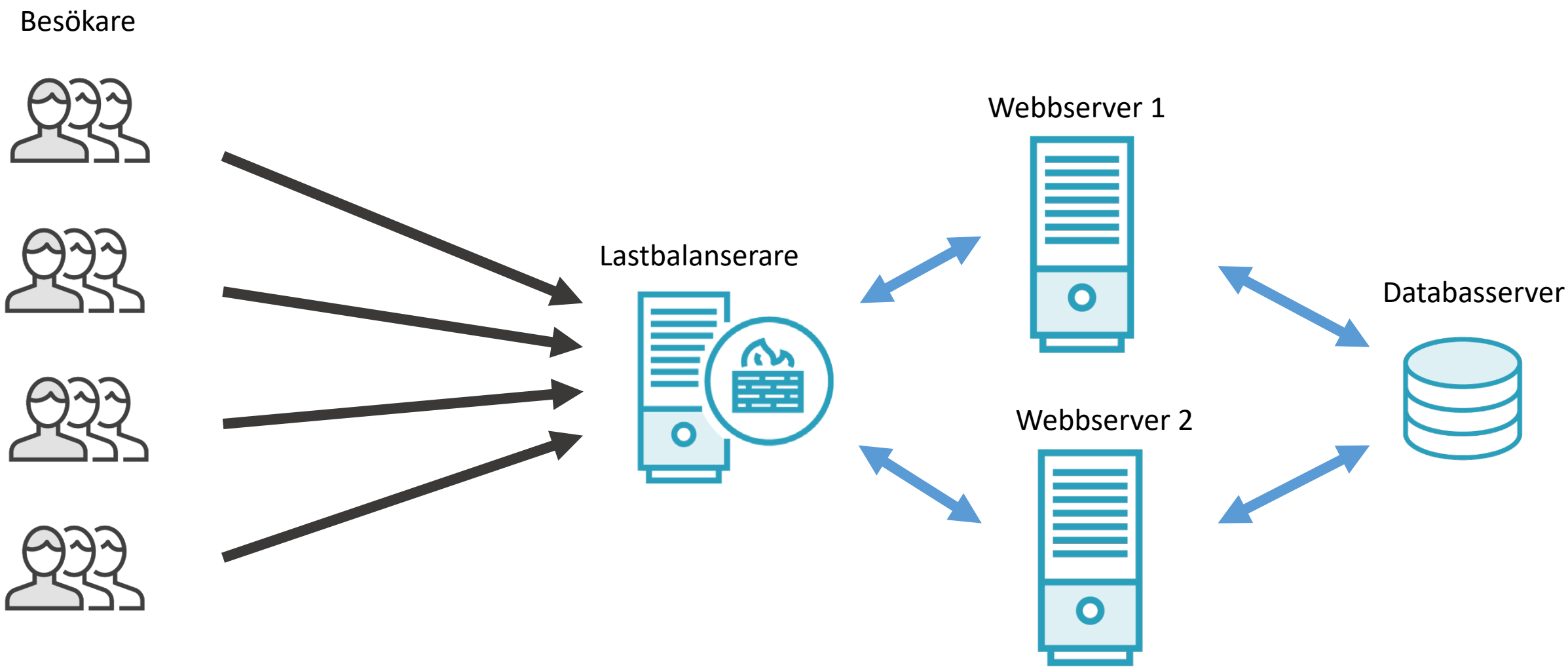
Hur vet ASP.NET vems Session som tillhör vem?

Request.Cookies;

En Cookie som identifierar vem du är!



Vad händer i en lastbalanserad miljö?


Besökare
Lastbalanserare
Webbserver 1
Webbserver 2
Databasserver

EntityFramework

Vad är EntityFramework?

Object-**R**elational **M**apping (**ORM**)

```
Install-Package EntityFramework
```

Installera EntityFramework NuGet-paketet i datalagret och i MVC projektet

Code First

Bygg upp vår databas-struktur genom att definiera data-modeller först

fekberg

Vy-modell och Data-Modell

Använd ett separat data-lager!

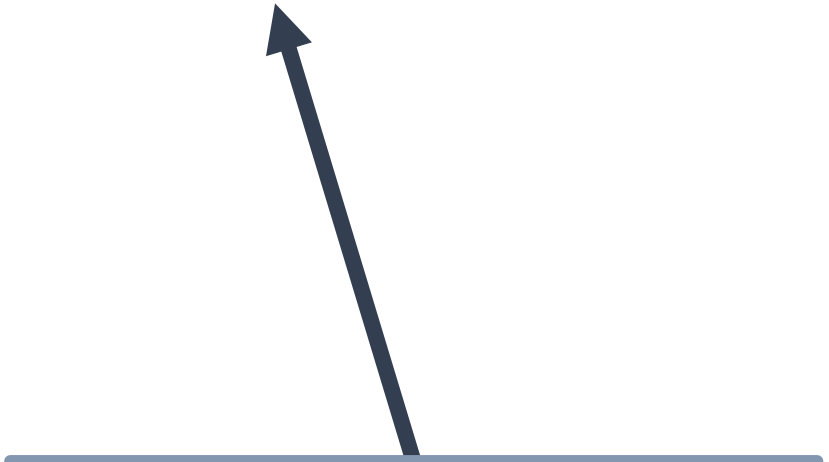
EntityFramework gör detta automatiskt till en PK

```
[Table("Person")]
public class Person
{
    public Guid Id { get; set; }

    [Index(IsUnique = true)]
    [MaxLength(30), MinLength(3)]
    public string Name { get; set; }

    [MaxLength(10), MinLength(5)]
    public string PhoneNumber { get; set; }

    public Person()
    {
        Id = Guid.NewGuid();
    }
}
```

```
public class MyDataContext : DbContext
{
    public DbSet<Person> People { get; set; }
}
```

```
[Table("Person")]
public class Person
```

```
using (var context = new MyDataContext())
```



Se till att vår context
Disposeas efter vi är Klara!

```
public IEnumerable<Person> All()
{
    using (var context = new MyDataContext())
    {
        return context.People.ToList();
    }
}
```

```
public void AddOrUpdate(Person person)
{
    using (var context = new MyDataContext())
    {
        var personToCreateOrUpdate =
            context.People.FirstOrDefault(p => p.Id == person.Id)
            ?? new Person();

        personToCreateOrUpdate.Name = person.Name;
        personToCreateOrUpdate.PhoneNumber = person.PhoneNumber;

        context.People.AddOrUpdate(personToCreateOrUpdate);
        context.SaveChanges();
    }
}
```

Hämta en redan existerande eller skapa ny

Uppdatera de fälten vi vill ändra på

Spara ändringarna

```
public void AddOrUpdate(Person person)
{
    using (var context = new MyDataContext())
    {

        ..........

    }
}
```

Varför använder vi inte ”person” direkt istället för att hämta ut det från databasen?

1. Vi har då full kontroll av det som uppdateras i databasen.

2. Vi behöver inte oroa oss för attached/detached eller dirty objekt

```
public void Delete(Guid personId)
{
    using (var context = new MyDataContext())
    {
        var toDelete = context.People.Single(p => p.Id == personId);

        context.People.Remove(toDelete);

        context.SaveChanges();
    }
}
```

Hur bör vi använda detta från ASP.NET MVC?

```
public ActionResult Index()
{
    IEnumerable<PersonViewModel> model;

    using (var context = new MyDataContext())
    {
        model = context.People.Select(person =>
            new PersonViewModel
            {
                Id = person.Id,
                Name = person.Name,
                PhoneNumber = person.PhoneNumber
            }
        );
    }

    return View(model);
}
```

Vy-modell, inte Data-modell!

Varför en vy-specifik modell?

Vy-modellerna innehåller ofta mer vy-specifik data!

fekberg

Håll ditt data-lager rent

```
public ActionResult Index()
{
    IEnumerable<PersonViewModel> model;

    using (var context = new MyDataContext())
    {
        model = context.People.Select(person =>
            new PersonViewModel
            {
                Id = person.Id,
                Name = person.Name,
                PhoneNumber = person.PhoneNumber,
                Awards = context.Awards.Where(award => award.AwardeeId == person.Id)
            }
        );
    }

    return View(model);
}
```

Knyta ihop det med Gallery-exempelkoden

fekberg

Migrations

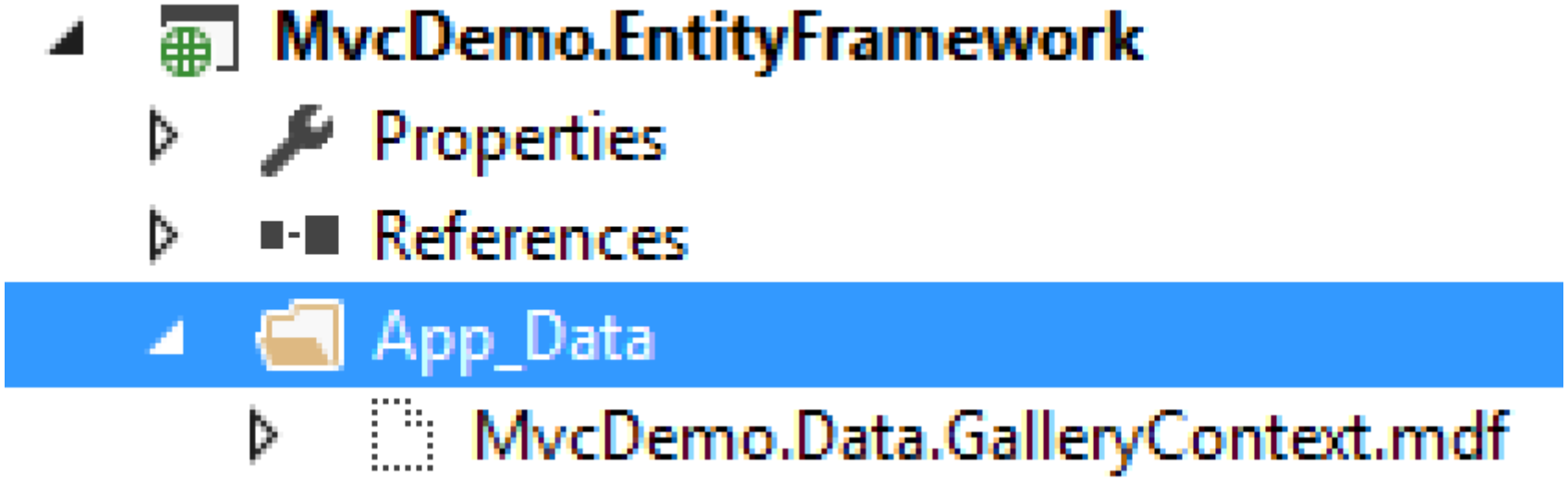
```
Enable-Migrations
```

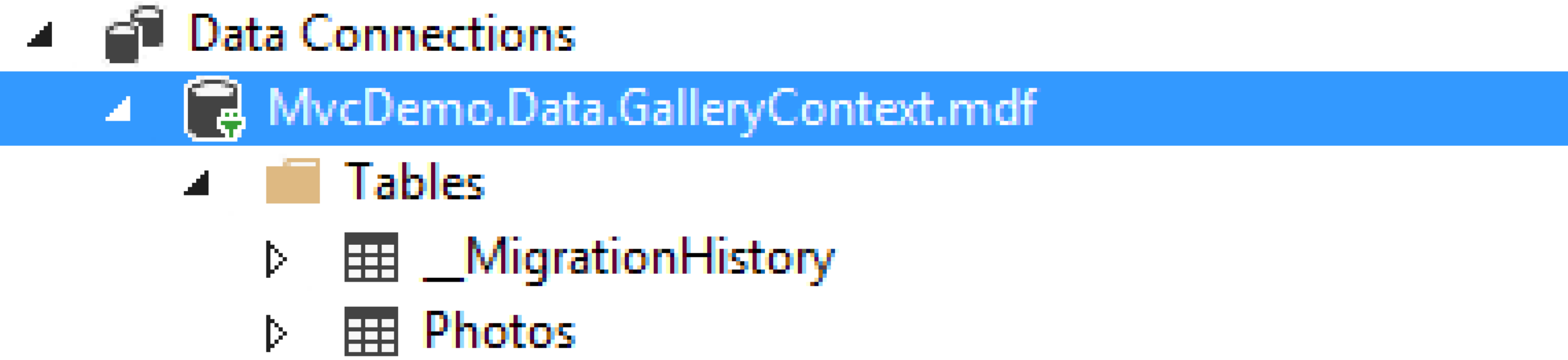
```
Add-Migration "Create Photo"
```

```
public class GalleryContext : DbContext
{
    public DbSet<Photo> Photos { get; set; }
}
```

```
public partial class CreatePhoto : DbMigration
{
    public override void Up()
    {
        CreateTable(
            "dbo.Photos",
            c => new
                {
                    Id = c.Guid(nullable: false),
                    Name = c.String(),
                    Filename = c.String(),
                })
            .PrimaryKey(t => t.Id);

    }

    public override void Down()
    {
        DropTable("dbo.Photos");
    }
}
```

```
Enable-Migrations –EnableAutomaticMigrations
```


MvcDemo.EntityFramework
Properties
References
App_Data
MvcDemo.Data.GalleryContext.mdf


Data Connections
MvcDemo.Data.GalleryContext.mdf
Tables
__MigrationHistory
Photos

dbo.Photos [Data]

Max Rows: 1000

Id	Name	Filename
a6d1088d-ec14-49af-a03d-697b38076e63	Sunrise	sunrise.jpg
90ca168f-8abf-4143-9a45-8e4f74ddd598	Sunset 2	sunset.jpg
47b7017d-5f9a-446e-99d3-b11aeb0d858b	Testbild	20160810_152309.jpg
NULL	*NULL*	*NULL*

Gallery/Index Title

localhost:58948/Gallery

Detta kommer från Index.cshtml

Ditt IP: ::1 (Visar ditt IP)
QueryString: (Visar allt efter ? i adressfältet)
Raw URL: /**Gallery** (Visar URLen som du skrivit in, exklusive hostname)

Nu laddar vi in en Partial View!

Ladda upp en ny bild

Choose File No file chosen Ladda upp

Nu tittar vi på en lista av bilder!

Sunrise

Ta bort

Sunset 2

Ta bort

```
<div id="result">
    @foreach (var photo in Model)
    {
        <div class="photo" style="float: left; padding: 20px;">
            <h3>@photo.Name</h3>
            <img src="/Uploads/@photo.Filename" height="250" />
            <div>
                @Ajax.ActionLink("Ta bort", "Delete", new { id = photo.Id },
                new AjaxOptions
                {
                    HttpMethod = "POST",
                    UpdateTargetId = "result",
                    Confirm = $"Är du säker på att du vill ta bort {photo.Name}?"
                })
            </div>
        </div>
    }
</div>
```

Hur lätt är det för oss att byta ut EntityFramework mot något annat nu?

Hur går vi från ”In-Memory” till EntityFramework?

När skapas vår databas?

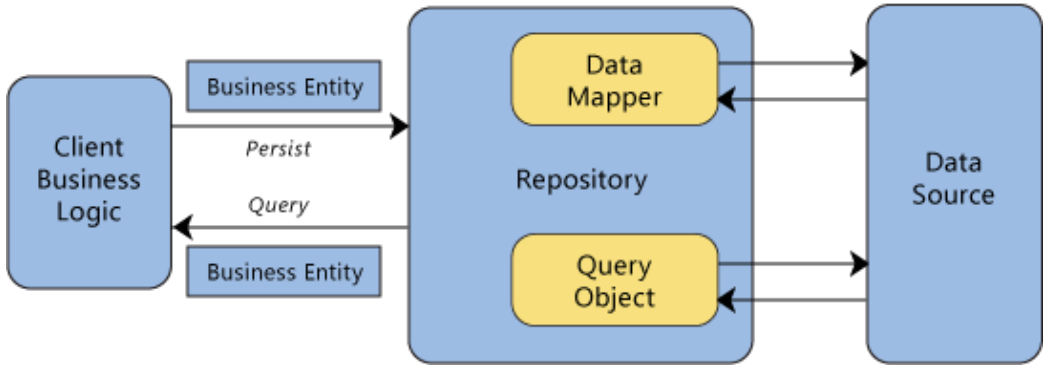
Glöm inte `using`

- **ADO.NET:** Execute Scalar "IF db_id(N'MvcDemo.Data.GalleryContext') I
- **ADO.NET:** Execute NonQuery "create database [MvcDemo.Data.Galler
- **ADO.NET:** Execute NonQuery "if serverproperty('EngineEdition') <> 5 e
- **ADO.NET:** Execute NonQuery "CREATE TABLE [dbo].[Photos] ([Id] [
- **ADO.NET:** Execute NonQuery "CREATE TABLE [dbo].[__MigrationHisto
- **ADO.NET:** Execute NonQuery "INSERT [dbo].[__MigrationHistory]([Mig
- **ADO.NET:** Execute NonQuery "INSERT [dbo].[Photos]([Id], [Name], [Fil
- **ASP.NET:** Redirect to "/Gallery/List"
- **ASP.NET:** GET "/Gallery/List"

Repository Pattern

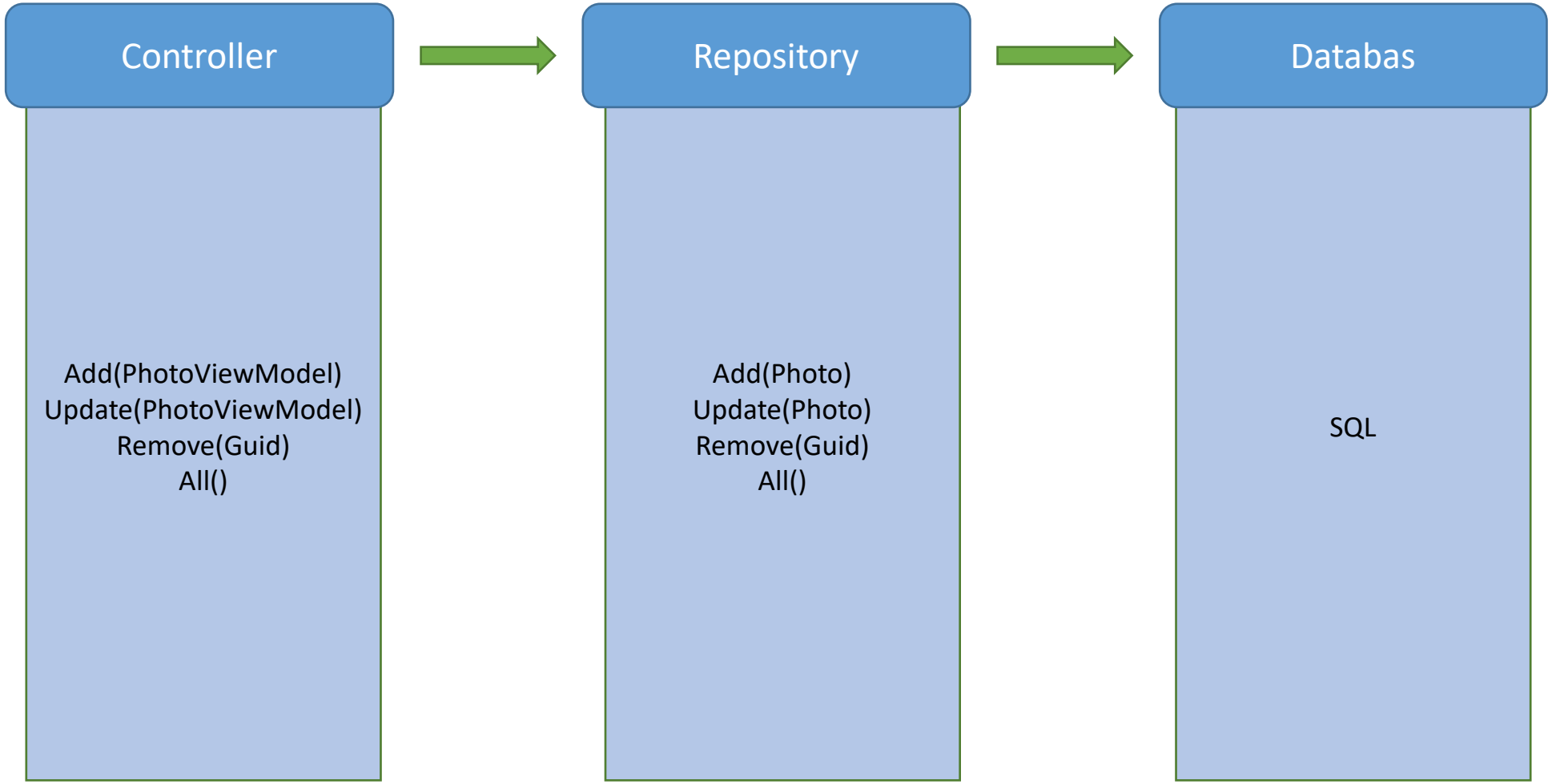
"The **repository pattern** is an abstraction. It's purpose is to reduce complexity and make the rest of the code persistent ignorant. As a bonus it allows you to write unit tests instead of integration tests."

Hur gör vi det enkelt för oss att jobba med ett data-lager utan att veta var datan lagras?



https://msdn.microsoft.com/en-us/library/ff649690.aspx


Controller
Repository
Databas


Controller
Repository
Databas
Add(PhotoViewModel)
Update(PhotoViewModel)
Remove(Guid)
All()
Add(Photo)
Update(Photo)
Remove(Guid)
All()
SQL

Controller

```
var photo = new Photo
{
    Id = Guid.NewGuid(),
    Filename = file.FileName,
    Name = model.Name
};
var repository = new PhotoRepository();

repository.Add(photo);
```

Repository

```
public void Add(Photo photo)
{
    using (var context = new GalleryContext())
    {
        context.Photos.Add(photo);
        context.SaveChanges();
    }
}
```

Databas

Varför?

Controller

```
var photo = new Photo
{
    Id = Guid.NewGuid(),
    Filename = file.FileName,
    Name = model.Name
};
var repository = new PhotoRepository();

repository.Add(photo);
```

Repository

```
public void Add(Photo photo)
{
    photosInMemory.Add(photo)
}
```

Testbarhet!

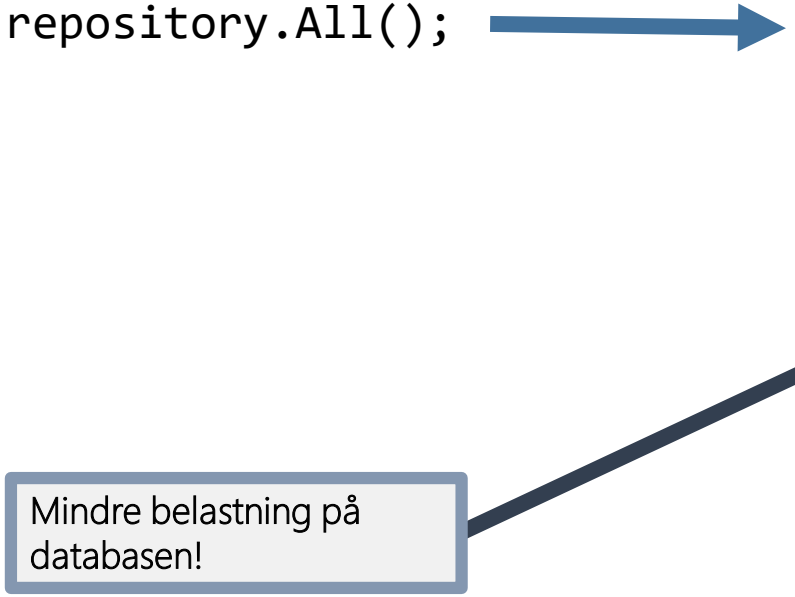
Enkelt att lägga in cache-strategi

Controller

```
var repository = new PhotoRepository();

var allPhotos = repository.All();
```

Repository

```
private static IEnumerable<Photo> photoCache;

public IEnumerable<Photo> All()
{
    if (photoCache != null &&
        photoCache.Any())
    {
        return photoCache;
    }

    using (var context = new GalleryContext())
    {
        photoCache = context.Photos.ToList();
    }

    return photoCache;
}
```



Hur ligger vi till?

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

Labb

Bildgalleri

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

Fortsätt med Bildgalleriet

- Lagra data i en databas och använd EntityFramework
- Använd ett separat data-lager med data-modeller
- Era vyer ska inte gå direkt mot modellerna som används i datalagret
- Bilder ska sparas på hårddisken, inte i databasen

mail@filipekberg.se

fekberg