# Persistens med "repository pattern"



Linneuniversitetet Kalmar ASP.NET MVC (1DV409)

#### Upphovsrätt för detta verk

Detta verk är framtaget i anslutning till kursen ASP.NET MVC vid Linnéuniversitetet.

#### Du får använda detta verk så här:

Allt innehåll i detta verk av Mats Loock, förutom Linnéuniversitetets logotyp och symbol, är licensierad under:



Creative Commons Erkännande-IckeKommersiell-DelaLika 2.5 Sverige licens. <a href="http://creativecommons.org/licenses/by-nc-sa/2.5/se/">http://creativecommons.org/licenses/by-nc-sa/2.5/se/</a>

#### Det betyder att du i icke-kommersiella syften får:

- kopiera hela eller delar av innehållet
- sprida hela eller delar av innehållet
- visa hela eller delar av innehållet offentligt och digitalt
- konvertera innehållet till annat format
- du får även göra om innehållet

Om du förändrar innehållet så ta inte med Linnéuniversitetets logotyp och symbol i din nya version!

Vid all användning måste du ange källan: "Linnéuniversitetet – ASP.NET MVC" och en länk till <a href="https://coursepress.lnu.se/kurs/aspnet-mvc">https://coursepress.lnu.se/kurs/aspnet-mvc</a> och till Creative Common-licensen här ovan.

Linneuniversitetet Kalmar

#### Persistent födelsedata i en XML-fil

- Data ska läsas från en XML-fil med namnet birthdates.xml. XML-noderna, med namnet birthdate, ska översättas till Birthday-objekt, sorteras på antal dagar kvar till nästa födelsedag och sedan presenteras.
- Nya Birthday-objekt ska kunna skapas och sparas i XML-filen.

```
<?xml version="1.0" encoding="utf-8"?>
<br/>
<br/>
dirthdates>
   <br/>
<br/>
dirthdate>
     <name>Ellen Nu</name>
     <date>2010-01-01T00:00:00</date>
    </birthdate>
    <br/>
<br/>
dirthdate>
      <name>Nisse Hult</name>
      <date>1967-02-11T00:00:00</date>
     </birthdate>
     <br/>
<br/>
dirthdate>
       <name>Carl Gustaf</name>
       <date>1946-04-30T00:00:00</date>;
      </birthdate>
    </birthdates>
```

## Nästa födelsedag!

- Ellen Nu kommer att fylla 5 år på en torsdag om 52 dagar.
- Nisse Hult kommer att fylla 48 år på en onsdag om 93 dagar.
- Carl Gustaf kommer att fylla 69 år på en torsdag om 171 dagar.

Lägg till födelsedag

Linnéuniversitetet Kalmar Vanyō ASP.NET MVC (1DV409)

#### Hur läsa data från en XML-fil?

- ✓ LINQ to XML är en teknik för att skapa, skriva och läsa XML-data.
- ✓ När en XML-fil har laddats in i ett XDocument-objekt är det enkelt att ställa frågor.

```
public ActionResult Index()
                              var path = Server.MapPath("~/App_Data/Birthdates.xml");
                              var model = (from birthdate in doc.Descendants("birthdate")
version="1.0" encoding
                                       select new Birthday
hdates>
                                            Name = birthdate.Element("name").Value,
                                            Birthdate = DateTime.Parse(birthdate.Element("date").Value)
irthdate>
<name>Ellen Nu</name>
<date>2010-01-01700:00
                                        }).OrderBy(b => b.DaysUntilNextBirthday).ToList();
</birthdate>
<br/>
<br/>
date>
  <name>Nisse Hult</nam
                                return View(model);
  <date>1967-02-11T00:6
 </birthdate>
  <br/>
<br/>
dirthdate>
    <name>Carl Gustaf</n
     <date>1946-04-30T0
    </birthdate>
  </birthdates>
```

#### Hur skriva data till en XML-fil?

✓ För att lägga till en ny nod i XML-trädet laddas hela XML-filen, en ny nod skapas som läggs till rotnoden och XML-trädet sparas.

```
[ValidateAntiForgeryToken]

[ValidateAntiForgeryToken]

[ValidateAntiForgeryToken]

[ValidateAntiForgeryToken]

[ValidateAntiForgeryToken]

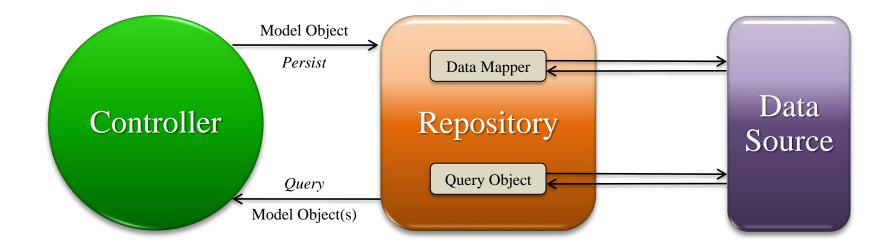
[ValidateAntiForgeryToken]
                                                     var path = Server.MapPath("~/App_Data/Birthdates.xml");
                                                 if (ModelState.IsValid)
                                                      var doc = XDocument.Load(path);
version="1.0" encoding="utf-8"?>
                                                                new XElement("name", birthday.Name),
                                                                new XElement("date", birthday.Birthdate));
                                                           new XElement("birthdate",
                                                       var element =
hdates>
irthdate>
<name>Ellen Nu</name>
<date>2010-01-01T00:00:00</date>
                                                       doc.Root.Add(element);
                                                       return RedirectToAction("Index");
</birthdate>
<br/>
<br/>
date>
  <name>Nisse Hult</name>
   <date>1967-02-11T00:00:00</date>
  </birthdate>
                                                   return View();
  <br/>
<br/>
dirthdate>
     <name>Carl Gustaf</name>
      <date>1946-04-30T00:00:00</dat
    </birthdate>
   </birthdates>
```

#### Vad måste controllern veta?

- ✓ Retoriska(?) frågor...
  - Om sättet att lagra persistent data ändras är det då lämpligt att controllern måste känna till hur lagringen av Birthday-objekts data sker?
  - Ar det en fördel att kod, som har med hantering av persistent data att göra, kan komma att behöva dupliceras av en eller flera controller?
  - Underlättar det att skriva tester mot controllrar då controllrar tar hand om alla detaljer beträffande hanteringen av persistent data?
  - Förenklas underhållet av applikationen då...? Är det enklare att...? Kommer...?
- ✓ ...och svaret på samtliga frågor är: **NEJ!**
- ✓ Genom att låta controllern kommunicera med ett centrallager för data ("repository"), en klass för hantering av persistent data, istället för direkt med XML-filen kan koden som har med hantering av persistent data lyftas ut från controllern.

#### "Repository pattern"

✓ Ett centrallager ("repository") finns kod för hantering av persistent data gentemot datakällan.



## "Repository"-klass

- Klassen XmlRepository kapslar in all hantering av persistent data mot XML-filen.
- Via de publika metoderna

  GetBirthdays, InsertBirtday

  och Save kan en controller hämta,

  lägga till och spara Birthday-objekt.

```
Public class XmlRepository
                          Private static readonly string PhysicalPath;
                         private XDocument _document;
                         Private XDocument Document
                              return _document ?? (_document = XDocument.Load(Physicalpath));
                   static XmlRepository()
                      PhysicalPath = Path.Combine(
                          sicalPath = Path.Combine(
AppDomain.CurrentDomain.GetData("DataDirectory").ToString(),

"Birthdates.xml"):
              public IEnumerable<Birthday> GetBirthdays()
                 return (from birthdate in Document.Descendants("birthdate")
                              Name = birthdate.Element("name").Value,
                             Name = birthdate.Element("name").Value,
Birthdate = Daterime.Parse(birthdate.Element("date").Value)
                       }).OrderBy(b => DateTime.Parse(birthdate.Element("dr
b.DaysUntilNextBirthday).ToList();
      public void InsertBirthday(Birthday birthday)
           new XElement("birthdate",
new XElement("name", birthday.Name),
new XElement("date", birthday.Birthdate)));
public void Save()
  Document.Save(PhysicalPath);
```

### Den nya controllern

- Då all funktionalitet för hantering av persistent data är placerad i klassen XmlRepository behöver inte controllern längre ha någon kännedom om hur Birthday-objekten lagras.
- Det enda controllern behöver göra är att instansiera ett XmlRepositoryobjekt och använda de publika medlemmarna.

```
public class BirthdayController
                         repository = new XmlRepository();
            // GET: /Birthday/
           Public ActionResult Index()
             return View(_repository.GetBirthdays()):
        // GET: /Birthday/Create
       Public ActionResult Create()
          return View();
    // POST: /Birthday/Create
   [HttpPost]
  [ValidateAntiForgeryToken]
 Public ActionResult Create(Birthday birthday)
        repository.InsertBirthday(birthday)
        repository.save();
     return RedirectToAction("Index");
return View();
```

Linneuniversitetet Kalmar ASP.NET MVC (1DV409

#### Mer information

- ✓ .NET Language-Integrated Query for XML Data
  - http://msdn.microsoft.com/en-us/library/bb308960.aspx
- ✓ LINQ to XML
  - http://msdn.microsoft.com/en-us/library/bb387098(v=vs.110).aspx
- Using LINQ to XML (and how to build a custom RSS Feed Reader with it)
  - http://weblogs.asp.net/scottgu/archive/2007/08/07/using-linq-to-xml-and-how-to-build-acustom-rss-feed-reader-with-it.aspx
- ✓ Introduction to LINQ, Part 2: LINQ to XML
  - http://www.codeguru.com/csharp/csharp/net30/article.php/c13715
- ✓ Using LINQ to XML to query XML data
  - http://dotnet.dzone.com/articles/using-linq-xml-query-xml-data