

Bygga testbara webbplatser med ASP.NET MVC

Mikael Lundin

Inledning

- Utvecklare – använd ASP.NET MVC
- Olika ramverk för att uppnå testbarhet
- Hur man tar sig runt de problem som det innefattar

Varför testbarhet?

- Högre kvalitet



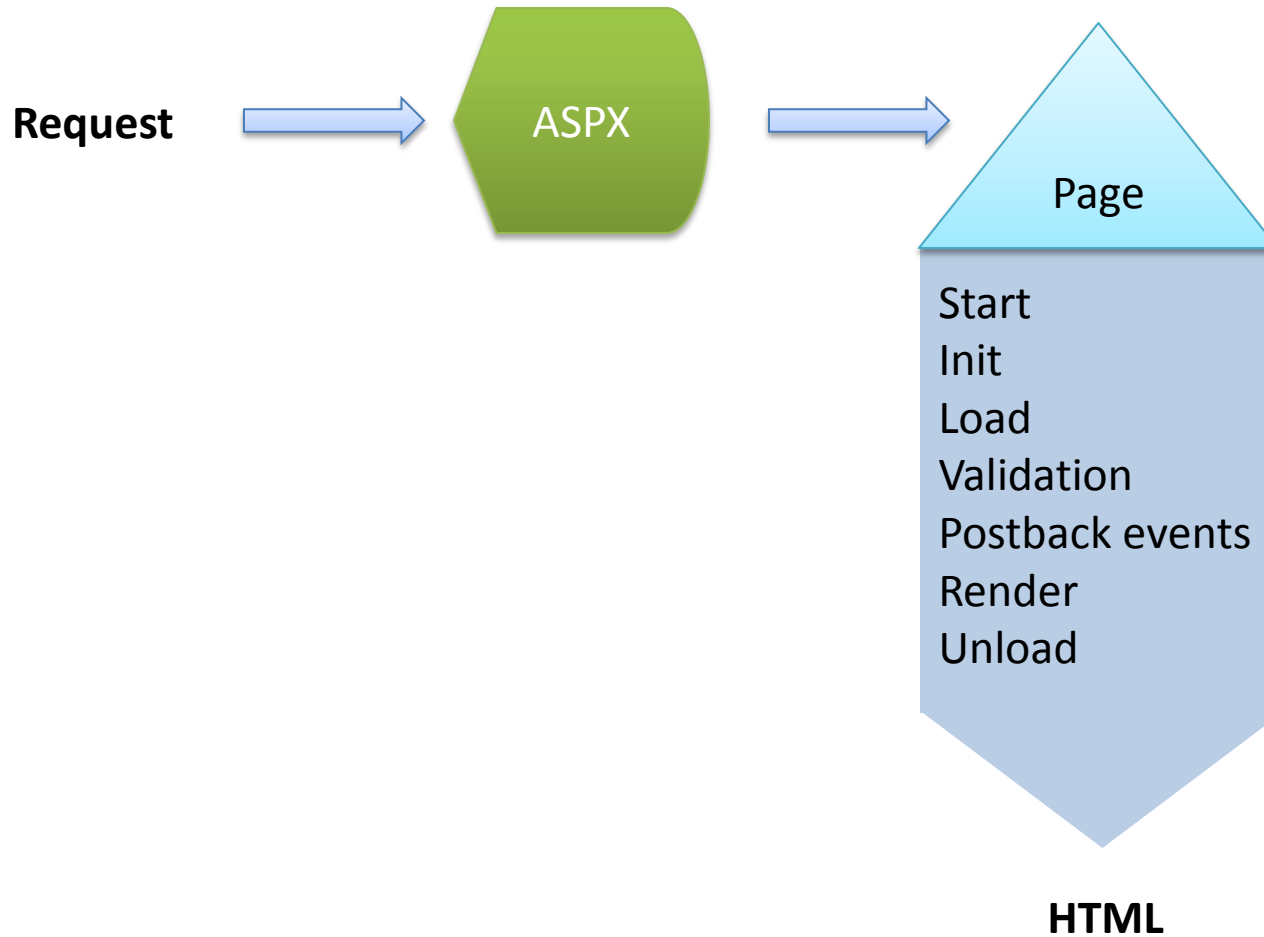
Testbarhet för förvaltning

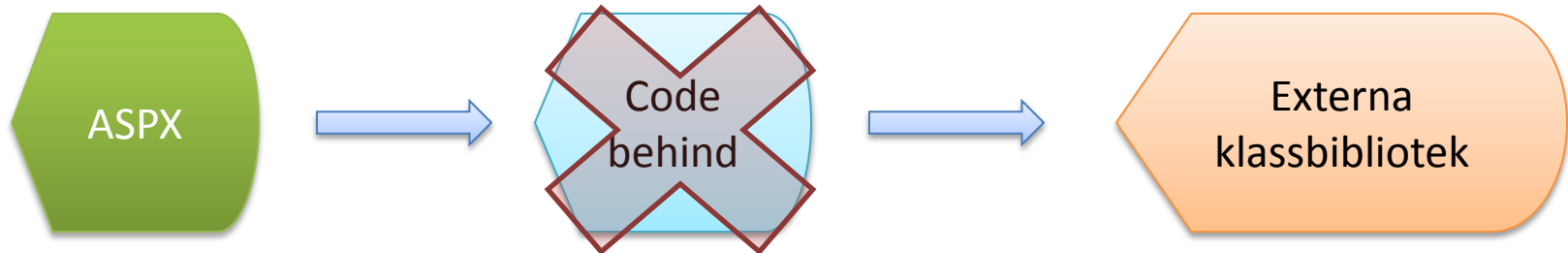
- Är systemet korrekt uppsatt på min maskin?
- Orsakar min förändring några bieffekter
- Testbar kod, lättare att förvalta

Testbarhet med WebForms

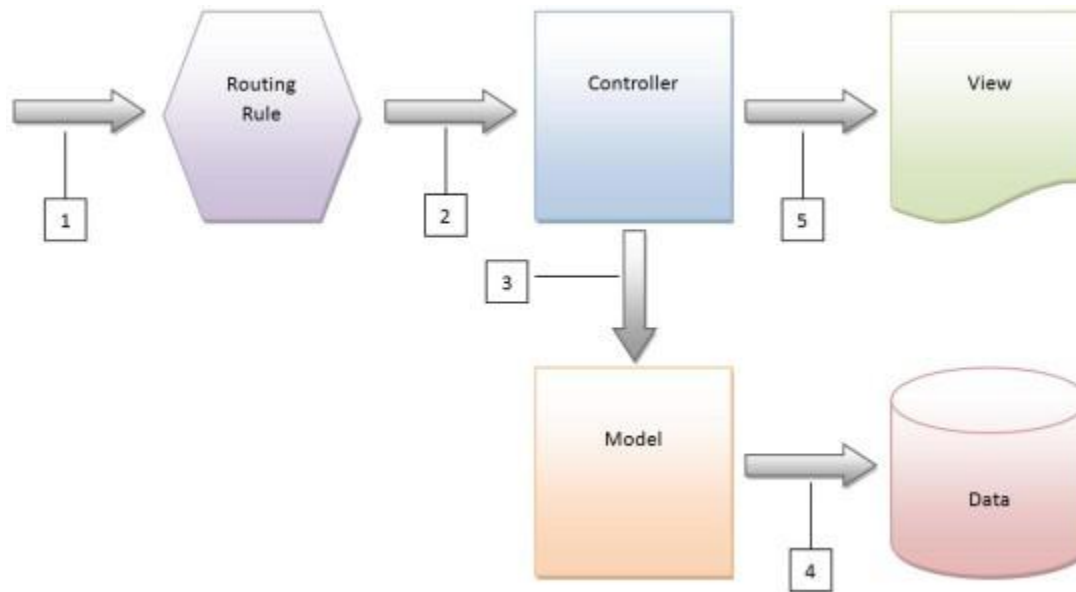
- Svårt att få till validerande XHTML
- ViewState
- HttpContext , HttpSessionState, HttpApplicationState
- Page life cycle
- Ingen kontroll över hur sidor skapas

ASP.NET WebForms

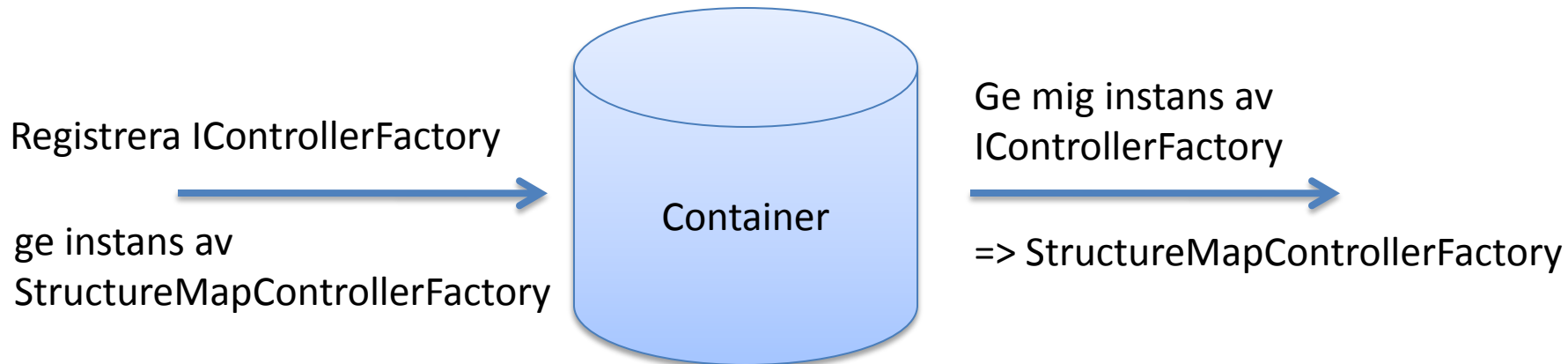




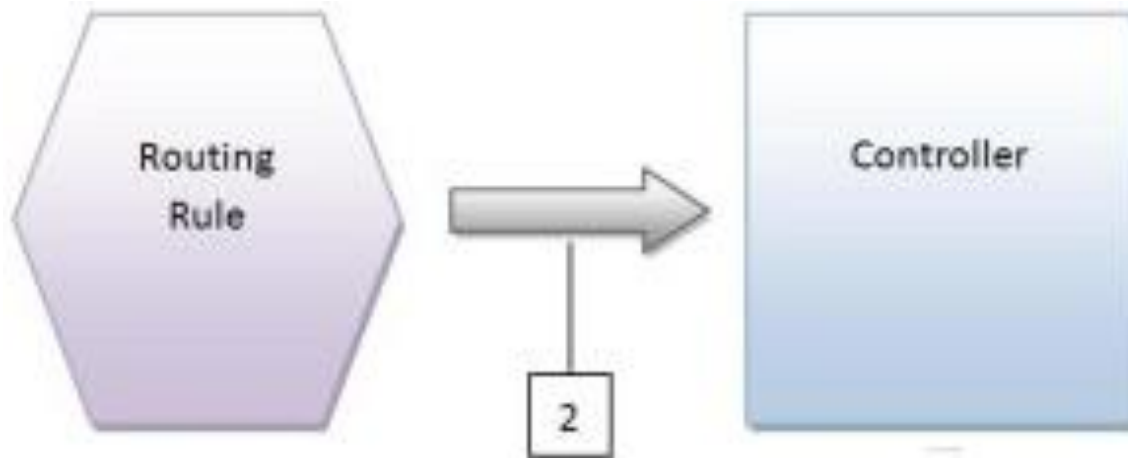
ASP.NET MVC



Dependency injection



Dependency injection



Dependency injection

```
public class Global : System.Web.HttpApplication
{
    public static void RegisterRoutes(RouteCollection routes) ...

    protected void Application_Start()
    {
        RegisterRoutes(RouteTable.Routes);
    }
}
```

Dependency injection

```
public class NewUserController : Controller
{
    private readonly IUserService userService;
    public NewUserController(IUserService userService)
    {
        this.userService = userService;
    }
}
```

Dependency Injection

- Unity
- StructureMap
- Castle Windsor
- Spring .NET

<http://mvcccontrib.codeplex.com/>

<http://www.valtech.se/dotnetdays/aspnetvmc/>

Hur gör man med sessionen?

HttpSimulator

```
[Test]
public void CannotCreateNewUserWhenAlreadyAuthenticated()
{
    /* Setup */
    var controller = new NewUserController();
    var newUser = new User { Username = "Kalle", Password = "Anka", Email = "kalle.anka@disney.se" };

    using (new HttpSimulator("/", @"c:\inetpub\").SimulateRequest())
    {

    }
}
```

Phil Haack: HttpSimulator

Skapa en state wrapper

```
public interface IWebContext
{
    IKeyValueState Session { get; }

    IKeyValueState Request { get; }

    IKeyValueState Application { get; }
}

public class HomeController : Controller
{
    private readonly IWebContext context;

    public HomeController(IWebContext context)
    {
        this.context = context;
    }
}
```

Exempel controller

```
public class HomeController : Controller
{
    private const string IsAuthenticated = "IsAuthenticated";
    private const string CurrentUserName = "CurrentUserName";
    private const string AuthenticatedMessage = "AuthenticatedMessage";

    private readonly IWebContext context;

    public HomeController(IWebContext context)
    {
        this.context = context;
    }

    }
}
```

Hur vi testar vår controller

```
[Test]
public void ShouldSetAuthenticatedMessageOnAuthenticatedUsers()
{
    /* Setup */
    const string UserName = "Mikael Lundin";

    var mock = new Mock<ControllerContext>();
    mock.Setup(c => c.HttpContext.Session["IsAuthenticated"]).Returns(true);
    mock.Setup(c => c.HttpContext.Session["CurrentUserName"]).Returns(UserName);

}
```

Hur testar vi vyn?

- <http://watin.sourceforge.net/>

Demo!

Hur testar vi vyn?

```
[Test]
public void ShouldHaveTopMenu()
{
    /* Setup */
    string url = Properties.Settings.Default.BaseUrl + "/Home";

    using (var browser = new IE(url))
    {
        /* Test */
        var topMenu = browser.Element("top-menu");

        /* Assert */
        Assert.That(topMenu.Exists, Is.True);
    }
}
```

<http://watin.sourceforge.net/>

<http://www.valtech.se/dotnetdays/aspnetvmc/>

WebForms MVP

- <http://webformsmvp.com/>
- EPiServerMVP
<http://github.com/joelabrahamsson/EPiServer-MVP>

Sammanfattning

- ASP.NET MVC underlättar för testbarheten
- Olika tekniker för att uppnå testbarhet – DI, mocka sessionen, WatiN

Föreläsningsmaterial

- <http://www.valtech.se/sv/valtechdays/aspnetmvc/>
- **Kontakt**
E-post: mikael.lundin@valtech.se
Twitter: @mikaellundin
Blogg: <http://mint.litemedia.se>