BEKK

ASP.NET VNEXT

Workshop

Skuret
Thor Kristian Valderhaug & Marius Løkketangen
14/4/2015

AGENDA

- INTRO TIL ASP.NET 5
- INTRO TIL ASP.NET MVC 6
- MER MVC 6 OG VISUAL STUDIO 15
- BOWER OG GRUNT

- Leaner, more modular, cross-platform, and cloud optimized.
- Open source.
- Inkluderer MVC 6.
- Dynamisk kompilering.
- ASP.NET Dependency Injection Framework.
- Bygd på .NET Core, men kjører også på .NET Framework.

- Navngitt etter "Project K".
- K – Verktøy for å kjøre ulike oppgaver fra kommandolinjen.
  - Nytt navn: DNX.
- KRE – K Runtime Environment.
  - Nytt navn: DNX.
- KVM – K Version Manager.
  - Nytt navn: DNVM - .Net Version Manager.
- KPM – K Package Manager (NuGet).

- Mappen ".k".
  - Ligger under "c:/users/USERNAME/.k".
  - NuGet-pakker vil ligge her og kunne brukes globalt for alle prosjekter på din maskin.
  - Runtime for .NET Core og .NET Framework vil også ligge her.

- .NET Core
  - Et subsett av .NET Framework, optimert for skyen.
  - Kan deployes med applikasjonen din slik at appen kan kjøre selvstendig, uavhengig av plattformen den kjøres på.
  - Core CLR ca 11 mb stor, Full CLR ca 200 mb.

- Man kan velge å bygge mot begge rammeverkene lokalt. Dette gjør at man kan luke ut eventuelle feil som kan komme av at man f.eks bygger mot Full CLR lokalt, mens man senere skal bygge mot Core CLR i et annet miljø.

- Filbasert prosjektsystem.
- For å legge til en fil til et prosjekt, putt filen i riktig mappe.
- Ingen global.asax, web.config eller .csproj-filer. All prosjekt-konfigurering skjer i json-filer og i Startup.cs.

- Global.json: Forteller hvor koden som skal kompileres ligger.
- Project.json: Inneholder prosjektinnstillinger.
  - Sier hvor statisk innhold ligger, som js og css-filer (default mappe heter wwwroot).
  - Definerer avhengigheter til NuGet-pakker.
  - Sier hvilke(n) CLR du vil bygge mot.
  - Definerer hvilke kommandoer du kan kjøre vha. "K".
  - Definerer andre automatiserte oppgaver (f.eks. "Pakk frontend-kode etter hvert bygg").
- Packages.json: Definerer avhengigheter til npm-pakker.
- Bower.json: Definerer avhengigheter til bower-pakker.

- Altså: Alle avhengigheter defineres i .json-filer.
- Andre konfigurasjonsvariabler som f.eks en DB-connection string kan også settes i en json-fil.
- Autocomplete for pakkehåndtering i Visual Studio 2015.

- Her skjer all konfigurering.
- public void ConfigureService(IServiceCollection services).
  - konfigurer og setter opp ønskede servicer.
- public void Configure(IApplicationBuilder app).
  - Setter opp request pipelinen.
- Egendefinert ASP.NET 5 middlevare.
  - Ta inn en RequestDelegat i konstruktøren.
  - Definere en metode public Task Invoke(HttpContext httpContext).
  - Lag en UseMidleware extension method på IApplicationBuilder.

Gjennomfør tutorial på

http://chad.tolkien.id.au/asp-net-vnext-ground-up-1-simplest-possible-thing/

Hvis man har gjennomført tingen vi sendte ut før påske kan man begynne på seksjonen project.json.

NB: bruk beta3 ikke beta2 versjonen av pakkene.

- No more Web Forms
- Tag Helpers

```
@model MyProject.Models.Product


@using (Html.BeginForm())
{
    <div>
        @Html.LabelFor(m => p.Name, "Name:")
        @Html.TextBoxFor(m => p.Name)
    </div>
    <input type="submit" value="Create" />
}
```

```
@model MyProject.Models.Product
@addtaghelper "Microsoft.AspNet.Mvc.TagHelpers"

<form asp-controller="Products" asp-action="Create"
method="post">
    <div>
        <label asp-for="Name">Name:</label>
        <input asp-for="Name" />
    </div>
    <input type="submit" value="Save" />
</form>
```

- View Components
  - Ansvar for en liten del lignende partial views
- Dependency Injection (ASP.NET)
- Routing
- Controller

- Dependency injection har blitt en “first-class citizen”
- Bruker “The Service Locator Pattern”
- En plass og registre alle rammeverk komponenter
- Kan enkel byttes ut med egen hvis man ønsker dette
  - Implementer public IServiceProvider ConfigureServices(IServiceCollection services) i Startup.cs
  - Virker ikke som om det har kommet ut noen DI enda

- Default routing oppfører seg som før by default, du trenger ikke å definere noe etter du har hentet mvc inn i pipelinen i Startup.cs
- Default routing kan overkjøres i Startup.cs ved å legge til routing-regler i metoden app.addMvc():

```
app.AddMvc(route => route.MapRoute(
                "default",
                "{controller=Home}/{action=Index)");
```

- Default routing oppfører seg som før by default, du trenger ikke å definere noe etter du har hentet mvc inn i pipelinen i Startup.cs
- Default routing kan overkjøres i Startup.cs ved å legge til routing-regler i metoden app.addMvc():

```
app.AddMvc(route => route.MapRoute(
                "default",
                "{controller=Home}/{action=Index)");
```

- Du har også muligheten til å sette en route på hver controller og action, uten å måtte definere noen generelle routing-regler i Startup.cs. Disse kan se ut som tidligere, eller være generiske:

```
[Route("/[controller]")]
public class HomeController : Controller {

        [Route("/[action]")]
        public IActionResult Index() { return View(); }
}
```

- Kan nå være POCO .
- Oppdaget basert på navnekonvensjon.
- MVC 6 = MVC + WebAPI.
- Bare en controller baseklasse som gir mulighet for å returnere views og json/xml.

Gjennomfør tutorial på
http://chad.tolkien.id.au/asp-net-vnext-mvc6-ground-up-2-plugging-in-mvc/

NB: bruk beta3 ikke beta2 versjonen av pakkene

Lag et WebApi med ASP.NET MVC 6:

Liten feil i tutorialen: UseWelcomePage() må flyttes/fjernes for at du skal få tilgang til api-et.

http://www.asp.net/vnext/overview/aspnet-vnext/create-a-web-api-with-mvc-6

- Alle er integrert i Visual Studio 2015.
- NPM
  - Node Package manager, håndterer pakker laget for Node.
  - Bower, Grunt og Gulp håndteres alle av npm.
- Bower
  - Package manager for frontend-pakker.
- Grunt og Gulp
  - Task-runners som f.eks. kan brukes til å automatisere minifisering av frontend-koden din.
  - Disse oppgavene kan settes opp fra «Task Runner Explorer» i Visual Studio 2015 til å kjøre automatisk f.eks. etter bygging.

Sette opp og bruke Grunt og Bower:

http://www.asp.net/vnext/overview/aspnet-vnext/grunt-and-bower-in-visual-studio-2015

- Mer MVC:
  - Leke videre med frontend for Todo-api (Bower, npm)
- Dependency injection
  - Bytte ut default IOC container med en egen
    - Hint: https://robinsedlaczek.wordpress.com/2014/11/22/dependency-injection-in-asp-net-vnext/
    - Hint2: legg til nuget feed https://www.myget.org/F/aspnetvnext/api/v2/

- Intro til ASP.NET 5: https://weblogs.asp.net/scottgu/introducing-asp-net-5
- Top 10 endringer: http://stephenwalther.com/archive/2015/02/24/top-10-changes-in-asp-net-5-and-mvc-6
- Hva kan vi forvente: http://developer.telerik.com/featured/expect-expecting-mvc-6/
- Pluralsight: http://www.pluralsight.com/courses/asp-dotnet-5-first-look
- Mer om View Components: http://www.dotnetthoughts.net/view-components-in-asp-net-mvc-6/

TAKK FOR OSS