DS3103 Webutvikling interface, model, controller

Rolando Gonzalez, 2022

Innhold

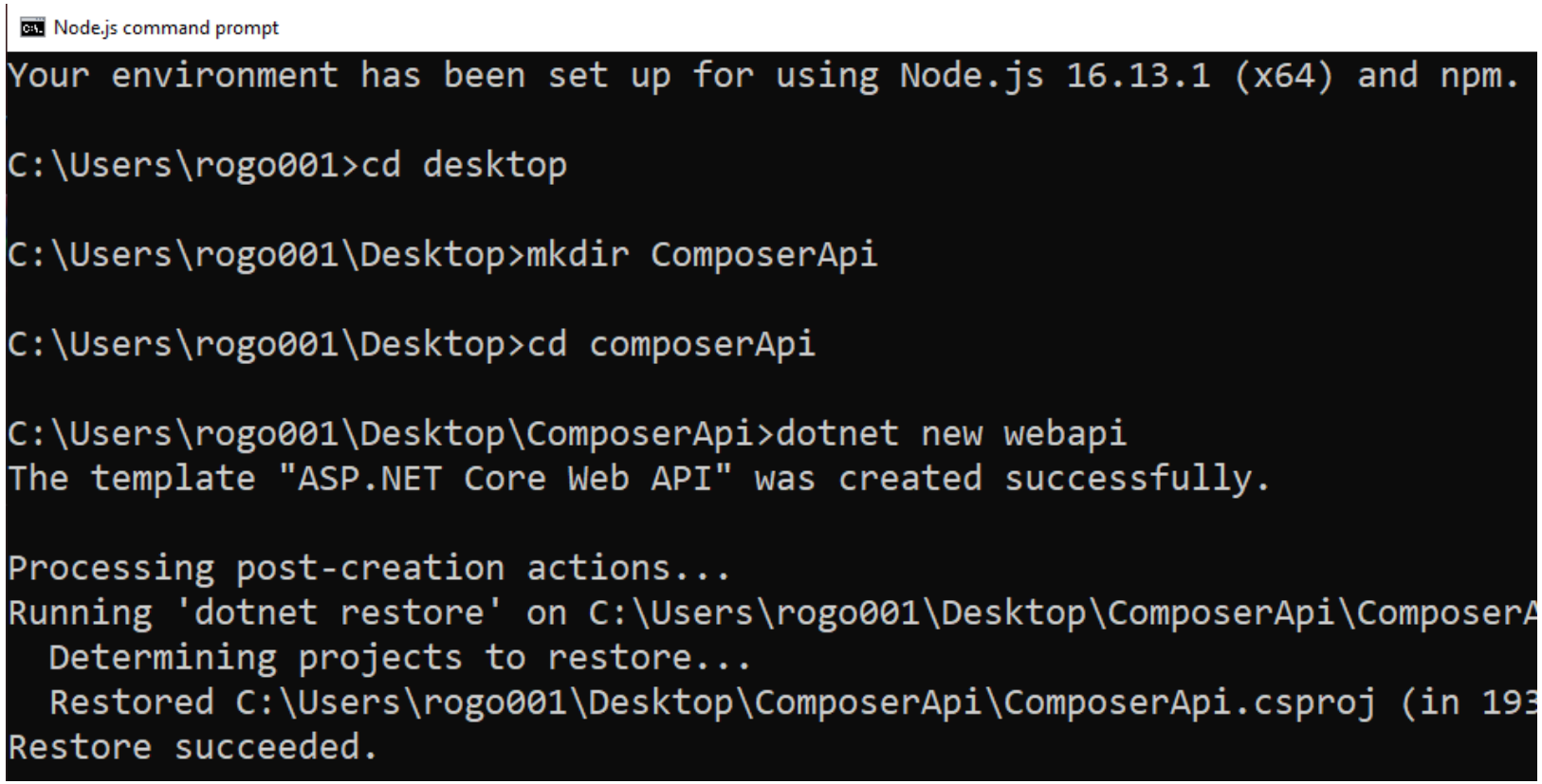
- Hensikten med denne slideserien
- Opprettelsen av boilerplate
- Interface
- Model
- Controller
- Kjøre i nettleser

Hensikten med denne slideserien

- Hensikten med denne slideserien er å gi et eksempel på hvordan man kan jobbe med C#-klasser og objekter i et Web Api.
- Det simulerer samtidig hvordan man jobber med klasser og objekter når man får informasjon fra en database.
- Når man jobber med databaser (SQL-database eller Dokumentdatabase som MongoDb) henter man inn informasjon fra databasen inn i Web Apiet. Informasjonen i databasen blir gjort om til objekter basert på klasser man har i Models-mappen som «mapper» til informasjonen i databasen.

Opprettelsen av boilerplaten

- Finner egnet sted å opprette Web Api boilerplate
- Oppretter mappe
- Går inn i mappen
- Kjører kommandoen for å skape boilerplate: dotnet new webapi

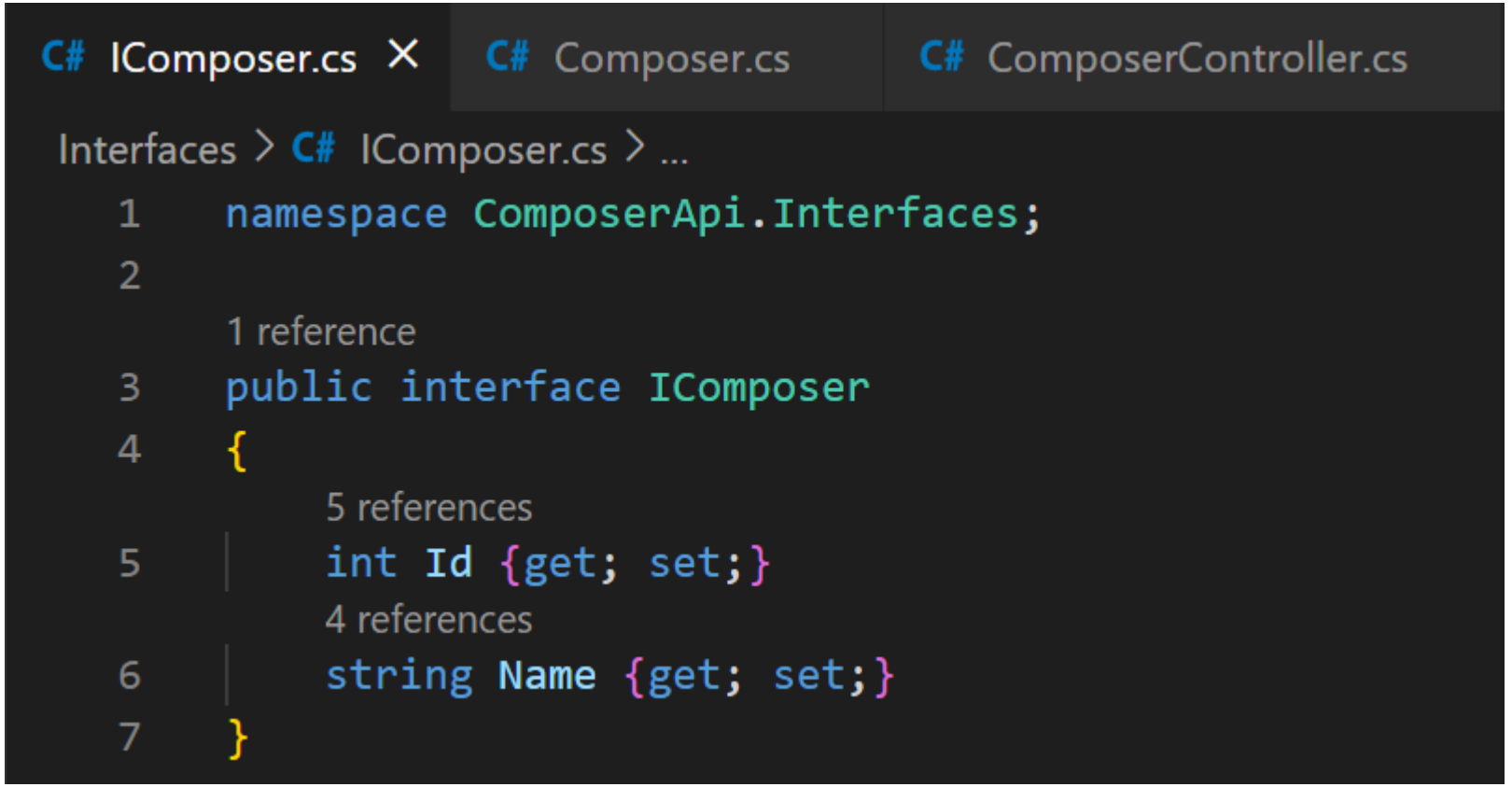


Interface

- Et interface angir hvordan en klasse skal se ut, en form for regel som skal følges når utviklere lager klasse av noe: “An interface defines a contract. Any class or struct that implements that contract must provide an implementation of the members defined in the interface.“ (Microsoft, 2022, https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/interface)

Icomposer.cs

- Setter namespace
- Deklarer public interface
- Definerer properties



Composer.cs

- Henter inn IComposer-tilgang
- Setter namespace
- Deklarerer klassen
- Setter properties som angitt i interfacet
- Setter default-verdi på Name i tilfelle det ikke settes

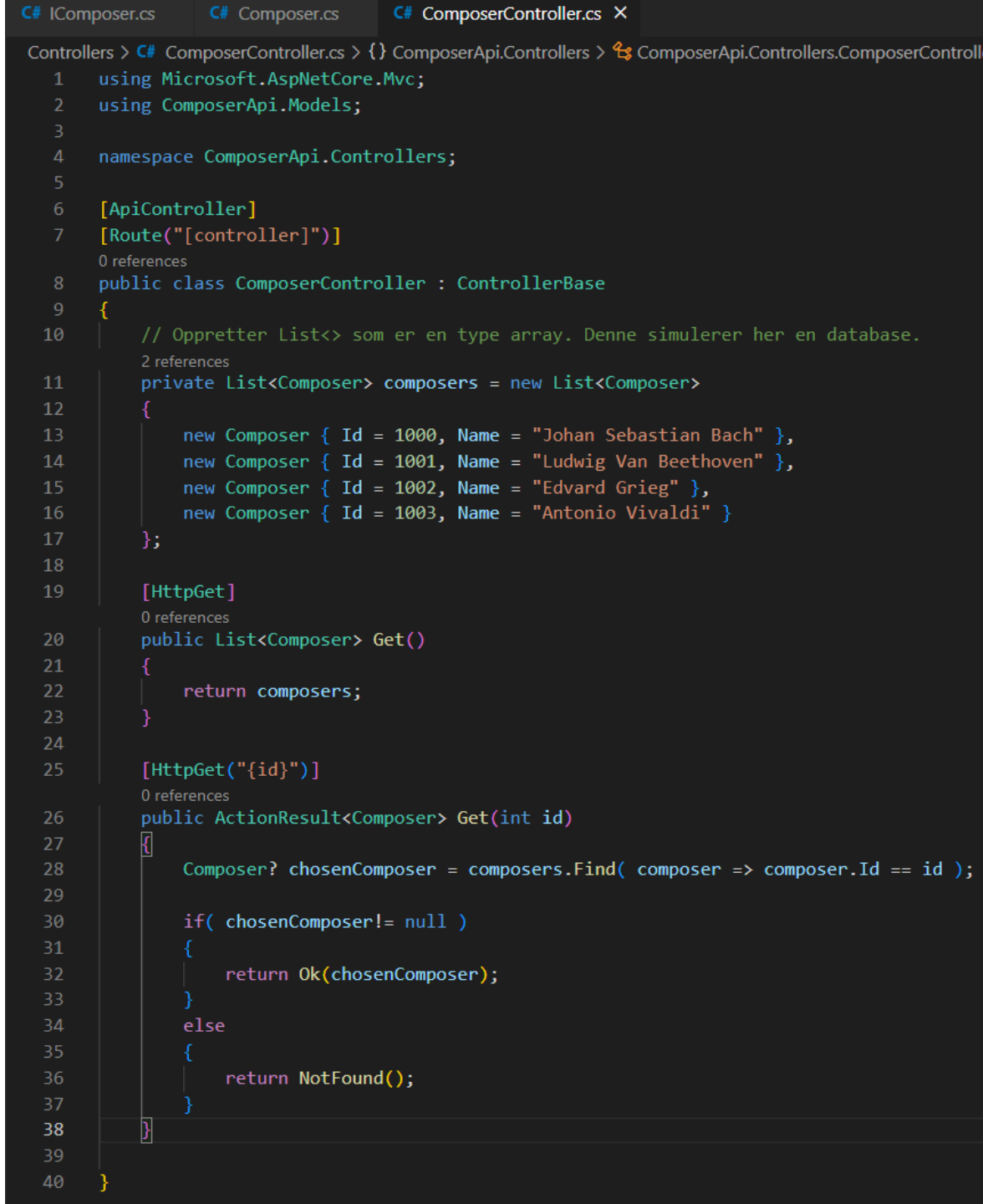
IComposer.cs | Composer.cs | ComposerController.cs

Models > Composer.cs > ...

```
1   using ComposerApi.Interfaces;
2
3   namespace ComposerApi.Models;
4
    9 references
5   public class Composer : IComposer
6   {
        5 references
7       public int Id {get; set;}
        4 references
8       public string Name {get; set;} = "Not set"; // "Not set" er default-verdi
9   }
```

ComposerController.cs

- Henter inn Mvc-namespacet
- Henter inn tilgang til Model-klassene
- Definerer namespace
- Dekorer med ApiController og setter Route
- Arver fra ControllerBase
- Oppretter List med Composer-objekter
- Oppretter 2 endepunkter: 1 for å hente alle, 1 for å hente etter id
- ActionResult gjør at vi kan returnere statuskoder (200, 400 osv.)



Kjøre i nettleser

- Kjører løsningen: dotnet watch run
- Går inn i url og taster inn url for de 2 endepunktene:
  - localhost:xxxx/composer
  - localhost:xxxx/composer/yyyy