# Introduction to MVC Tutorial - Building my first MVC application

**Introduction**

This tutorial will build on the practices and the concepts you were introduced to in the lecture.

**Required Software**

For this tutorial, you will need access to the following software applications:

- MS Explorer
- Visual Studio

**Reference**

Walther et al, 'ASP.NET MVC Framework Unleashed', SAMS

**Breakdown of Tutorial**

This tutorial consists of the following tasks:

You will be given a series of practical exercises, along with selected questions to consolidate your learning.

**Review questions**

| | |
|---|---|
| **Refreshers:** Recall from memory as much as possible to answer the following refresher questions. Then open the '*An Introduction to MVC Lecture*' presentation to help you develop and extend your answers. | |
| 💡 | You have built many Web applications and so you should be familiar with the structure of these applications. Explain these applications in terms of components, structure, flow of messaging, etc. Use simple diagrams to help you formulate your answers. |
| | |
| 💡 | How are MVC Web applications different from the ones you have built in the past? Again use simple diagrams to help you present your thoughts. |
| | |
| 💡 | Do you perceive any advantages with MVC Web applications? |
| | |
| 💡 | There are two approaches to using Entity Framework. One is Database First. The other is Code First. Describe each approach. Compare and contrast the two approaches. |
| | |

| | |
|---|---|
| | In this tutorial we will use Database First approach. |

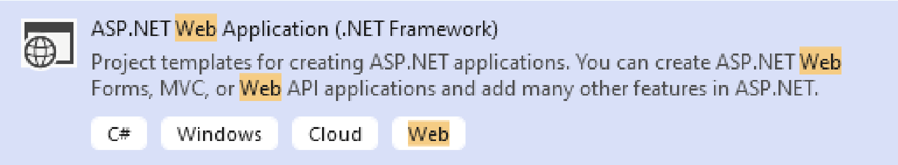
**Tutorial objectives**

The object of this exercise is to build our first MVC Web application. We will become familiar with the structure and components of an MVC Web application.

**Creating and hosting your first ASP.NET MVC application**

We need to adopt a different approach to building our MVC applications. Developing Web Applications in general follows a particular approach. That is application development and testing is carried out before the application is hosted on the web Server.  Developing MVC applications also follows the same pattern.

_ Launch Visual Studio.
_ Create a new Project. In the Project Dialog opt for *ASP.NET Web Application (.NET*



ASP.NET Web Application (.NET Framework)

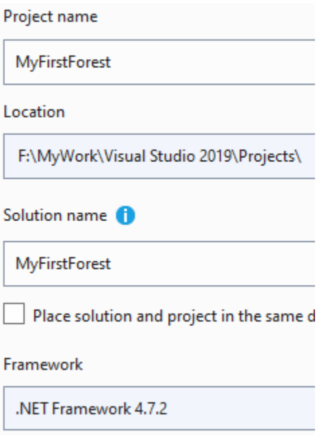
Project templates for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

*Framework)* Web-Template and name your project *MyFirstForest*.

_ Note that programming language defaults to *C#*.

_ Navigate to the location: "..\ *MyWork\Visual Studio 2019*\", create a new folder called "*Projects*" and store the project in:" .. *MyWork\Visual Studio 2019\Projects*\", opt for MVC, and click *Create*.

_ Study the structure and what comes readily shipped with the project.



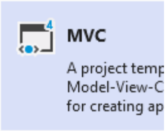
Project name: MyFirstForest

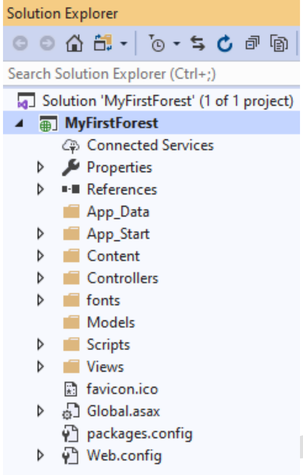
Location: F:\MyWork\Visual Studio 2019\Projects\

Solution name: MyFirstForest

□ Place solution and project in the same d

Framework: .NET Framework 4.7.2





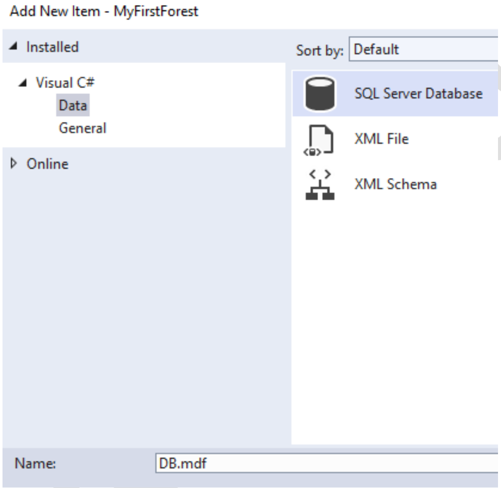
It is a good idea to create the unit test project as creating it later is quite a complex process. However, we are not going to do any unit testing. Unit testing is a subject in its own right.

## Creating the database

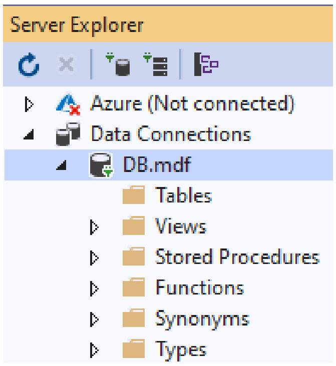
Locate *Server Explorer*. You may have to use the *View* menu to open it.

_ Right click on *App_Data* folder and *Add a New Item*, a *SQL Server Database*, and name it *DB.mdf*

_ Double-Click DB.mdf, it opens in Server Explorer. You can develop the database within Server Explorer, for example add tables to the database, browse through the tables and stored procedures in your database



Name: DB.mdf



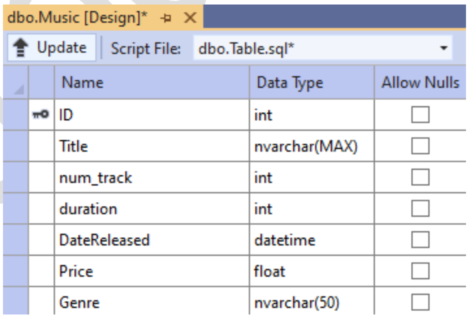
## Building a table in SQL server and inserting data into the table from Within Visual Studio

Having connected to our database we now want to create our database for Forest case study.

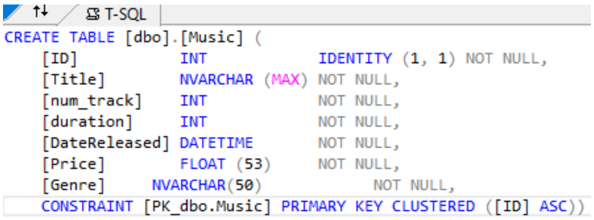
Create the ***Music*** table, working within the *Server Explorer*:

_Right_click on *Tables*, *Add New Table*

_Prepare the Table in Table Design View



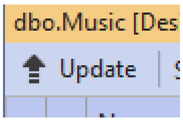
| Name | Data Type | Allow Nulls |
|---|---|---|
| ID | int | □ |
| Title | nvarchar(MAX) | □ |
| num_track | int | □ |
| duration | int | □ |
| DateReleased | datetime | □ |
| Price | float | □ |
| Genre | nvarchar(50) | □ |

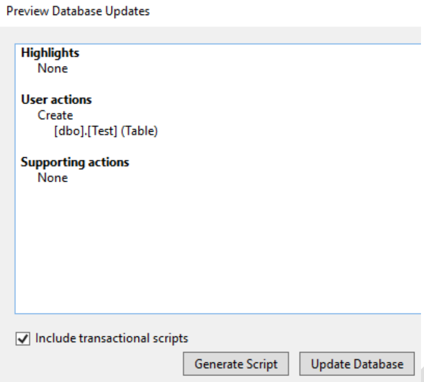


```
CREATE TABLE [dbo].[Music] (
    [ID]           INT            IDENTITY (1, 1) NOT NULL,
    [Title]        NVARCHAR (MAX) NOT NULL,
    [num_track]    INT            NOT NULL,
    [duration]     INT            NOT NULL,
    [DateReleased] DATETIME       NOT NULL,
    [Price]        FLOAT (53)     NOT NULL,
    [Genre]     NVARCHAR(50)          NOT NULL,
    CONSTRAINT [PK_dbo.Music] PRIMARY KEY CLUSTERED ([ID] ASC))
```

_ At the top of the *Design* window, click *Update*, and *Update Database*

_Within the *Server Explorer*, click on *Tables*, *Refresh* (↻), and check that the *Music* table has been created




Preview Database Updates

**Highlights**
None

**User actions**
Create
[dbo].[Test] (Table)

**Supporting actions**
None

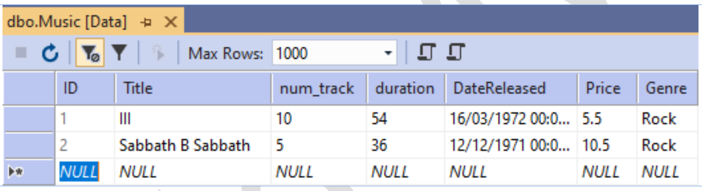
☑ Include transactional scripts

_ Click on *Update Database*.

_ ↻ Click Refresh to see the table.

To add records to the table;

_ Right click on the table and *Show Table Data*
_ Add a few records to the table – Use the *Tab* key to move from cell to cell and to new record



| | ID | Title | num_track | duration | DateReleased | Price | Genre |
|---|---|---|---|---|---|---|---|
| | 1 | III | 10 | 54 | 16/03/1972 00:0... | 5.5 | Rock |
| | 2 | Sabbath B Sabbath | 5 | 36 | 12/12/1971 00:0... | 10.5 | Rock |
| ▸* | NULL | NULL | NULL | NULL | NULL | NULL | NULL |

**Installing Entity Framework in the solution**

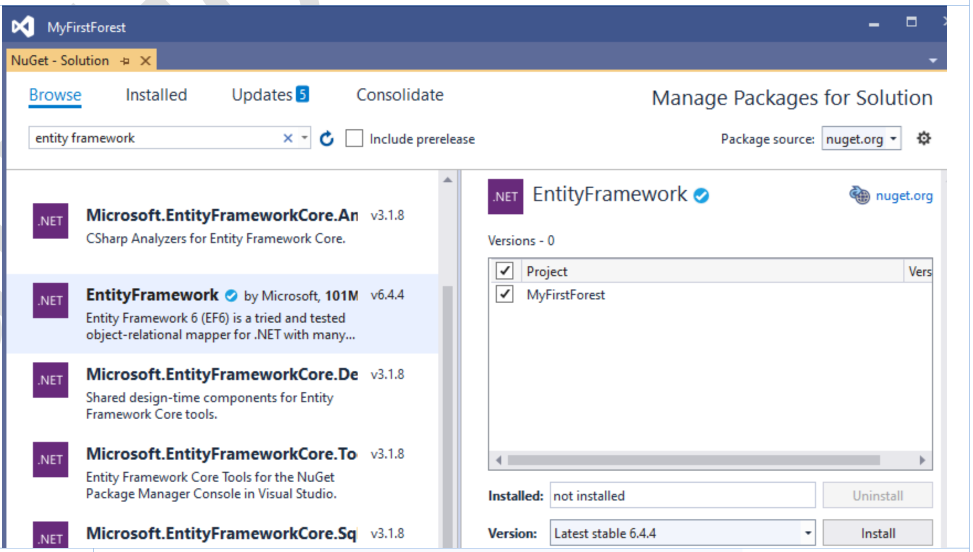
We use **Microsoft Entity Framework** because it is the recommended ORM for data access solution but at times we may also use **LINQ to SQL Classes**.

We must first build the entity framework assembly into the solution.

The assembly that contains Entity Framework is System.Data.Entity. Once installed, you will find this assembly under *References*.

Use the menu …

_ Tools – NuGet Package Manager – Manage NuGet Packages for Solution..
_ Browse for entity framework ..
_ opt for EntityFramework v6 ..
_ Tick Project and MyFirstForest ..
_ Click Install



entity framework ☐ Include prerelease Package source: nuget.org

Microsoft.EntityFrameworkCore.An v3.1.8
CSharp Analyzers for Entity Framework Core.

EntityFramework by Microsoft, 101M v6.4.4
Entity Framework 6 (EF6) is a tried and tested object-relational mapper for .NET with many...

Microsoft.EntityFrameworkCore.De v3.1.8
Shared design-time components for Entity Framework Core tools.

Microsoft.EntityFrameworkCore.To v3.1.8
Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.

Microsoft.EntityFrameworkCore.Sq v3.1.8

EntityFramework nuget.org

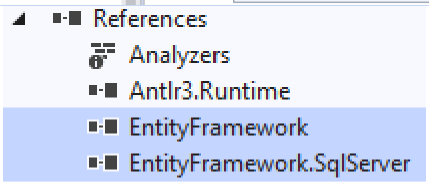
Versions - 0

☑ Project Vers
☑ MyFirstForest

Installed: not installed Uninstall

Version: Latest stable 6.4.4 Install

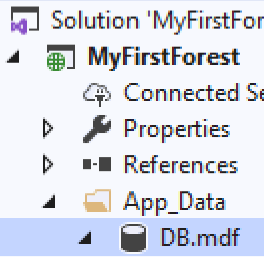
_ Examine the References folder and the two assemblies for Entity Framework

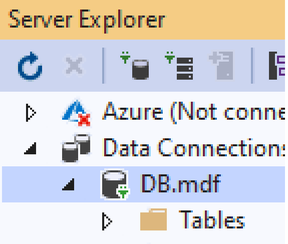


**Creating the Model**

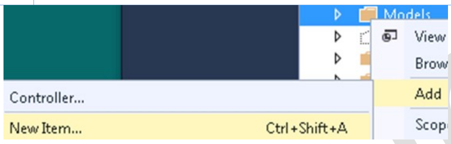
We use **Microsoft Entity Framework** because it is the recommended ORM for data access solution but at times we may also use **LINQ to SQL Classes**.

_ Double-Click on DB.mdf in App_Data folder
_ Ensure you see the database under Data Connections in Server Explorer

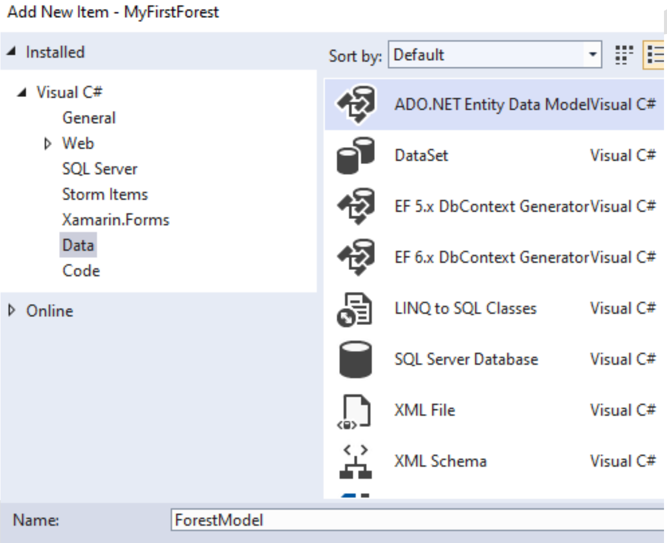

Solution 'MyFirstFor
MyFirstForest
Connected Se
Properties
References
App_Data
DB.mdf


Server Explorer
Azure (Not conne
Data Connection:
DB.mdf
Tables

_ Right-Click on the *Models* folder and *Add a New Item*.
_ Under the *Data* category of installed templates opt for *ADO.NET Entity Data Model*.
_ Name it *ForestModel*.
_ Click *Add* to create the model.



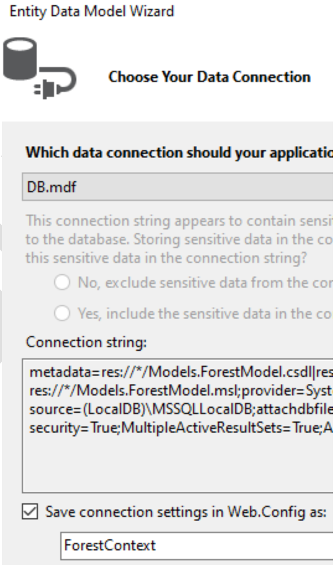
and


Add New Item - MyFirstForest

| | | |
|---|---|---|
| Installed | Sort by: Default | |
| Visual C# | ADO.NET Entity Data Model | Visual C# |
| General | DataSet | Visual C# |
| Web | EF 5.x DbContext Generator | Visual C# |
| SQL Server | EF 6.x DbContext Generator | Visual C# |
| Storm Items | LINQ to SQL Classes | Visual C# |
| Xamarin.Forms | SQL Server Database | Visual C# |
| Data | XML File | Visual C# |
| Code | XML Schema | Visual C# |
| Online | | |

Name: ForestModel

Run through the wizard:
_ Opt for EF Designer from database
_ Opt for the appropriate *Data Connection*
_ Ensure that *Connection String* is saved to the *Web.Config*.
_ Make sure *Genre* table is checked.


Entity Data Model Wizard

Choose Your Data Connection

Which data connection should your applicatio

DB.mdf

This connection string appears to contain sensi
to the database. Storing sensitive data in the co
this sensitive data in the connection string?

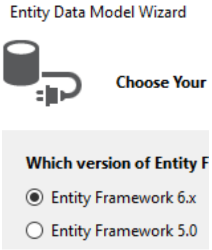
○ No, exclude sensitive data from the cor
○ Yes, include the sensitive data in the co

Connection string:

```
metadata=res://*/Models.ForestModel.csdl|res
res://*/Models.ForestModel.msl;provider=Syst
source=(LocalDB)\MSSQLLocalDB;attachdbfile
security=True;MultipleActiveResultSets=True;A
```

☑ Save connection settings in Web.Config as:

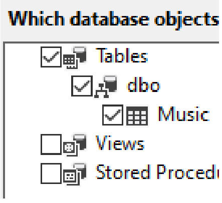
ForestContext


Entity Data Model Wizard

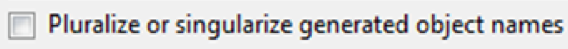
Choose Your

Which version of Entity F
● Entity Framework 6.x
○ Entity Framework 5.0

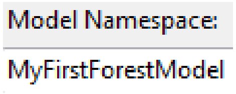
and


Which database objects
☑ Tables
☑ dbo
☑ Music
□ Views
□ Stored Proced

_ Ensure that you take the '*Plurize or singularize generated object names*' off.
_ Note the *Model Namespace*, change to *MyFirstForestModel*

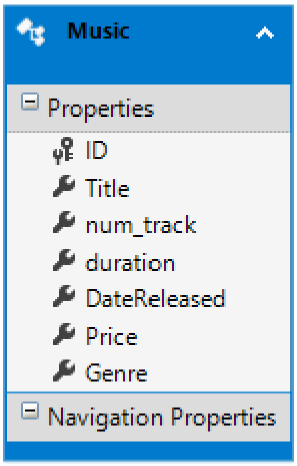

□ Pluralize or singularize generated object names

_ Click *Finish* to create the model.



Model Namespace:

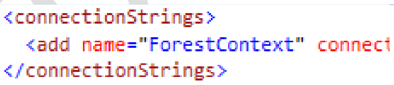
MyFirstForestModel



Working in Solution Explorer ..
_ Right-Click on the Solutin and Build the solution.

On inspection;

_ A new *Conncetion String* is written to the *Web.Config* file at the root of the project.



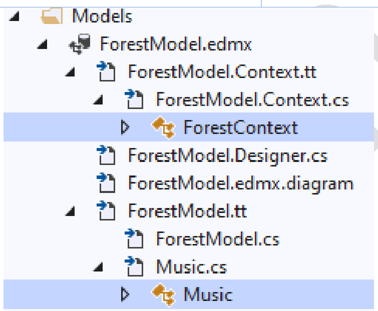
```
<connectionStrings>
  <add name="ForestContext" connect
</connectionStrings>
```

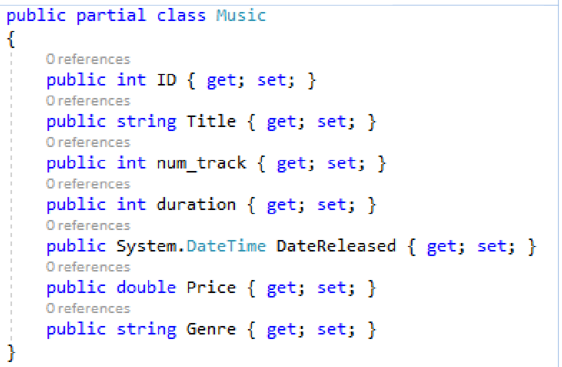
_ Two classes of interest are created:

  _ One is the *ForestContext* class. Open the class and inspect that it inherits *DbContext* class.

  _ One is the *Music* class.

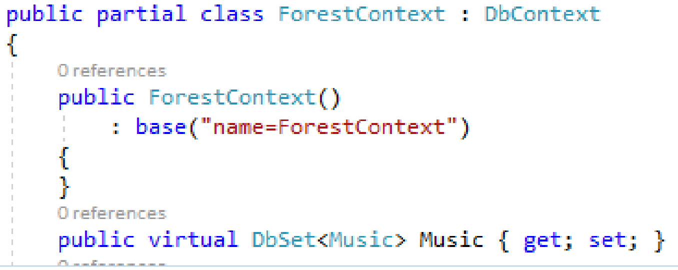
_ Note the two properties of this class that map to properties of *Music* table in your database.





```
public partial class Music
{
    0 references
    public int ID { get; set; }
    0 references
    public string Title { get; set; }
    0 references
    public int num_track { get; set; }
    0 references
    public int duration { get; set; }
    0 references
    public System.DateTime DateReleased { get; set; }
    0 references
    public double Price { get; set; }
    0 references
    public string Genre { get; set; }
}
```

_ Note the `ForestContext` class.
_ Note that this class inherits *DbContext* class.
_ Note that this class has the same name as your *Connection String*.
_ Note the property Music of type `DbSet<Music>`



```
public partial class ForestContext : DbContext
{
    0 references
    public ForestContext()
        : base("name=ForestContext")
    {
    }
    0 references
    public virtual DbSet<Music> Music { get; set; }
```

In order to create the model class in ASP.NET we make use of an Object Relational Mapping (ORM). ADO.NET Entity Framework is an implementation of ORM.

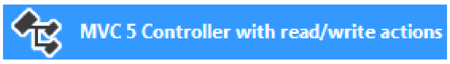
What is Object Relational Mapping (ORM)?

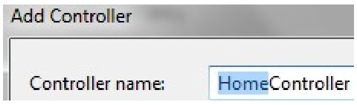
Do you have examples of ORM?

Note the *Music* class. *ADO.NET Data Entity Framework* has provided us with a class representation of the database table. What are the benefits of this? Why do we do it?

**Creating the HomeController**

_ Delete the *HomeController* from the project. You would find it in../*Controllers* folder.
_ Delete the ../*Views/Home* folder.
_ Right click on the Controllers folder and add a new controller with read/write actions.





Controller name: HomeController

Controller has a number of Controller *Actions*.

Inspect the controller and answer the following questions;

What class does the controller inherit from? What is the namespace for that class?

What is the naming convention for controllers?

What is the function of the *Index()* action?

What is the function of the *Details(int id)* action?

What is the function of the *Create()* action?

What is the function of the *Create(FormCollection collection)* action?

What is the function of the *Edit(int id)* action?

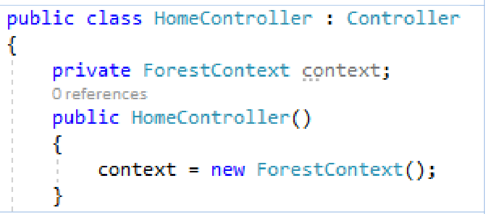
What is the function of the *Edit(int id, FormCollection collection)* action?

To work with the data, we would like to take advantage of the Data Model Class that we created using the *ADO.NET Data Entity Framework*, in implementing the action. *ForestContext* class (of type *DbContext* class) is the class that facilitates access to the database.

What is the namespace that *ForestContext* class is in?

Working on the *HomeController*;

_ Create an object of the *ForestContext* class.
_ Note that here we are using a constructor to create the object. Object can then be used in all methods with the *HomeController* class.

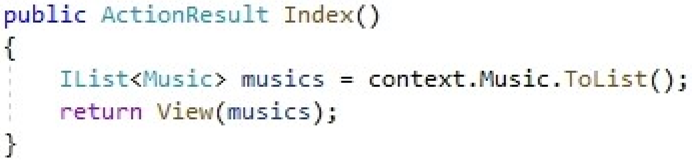


```
public class HomeController : Controller
{
    private ForestContext context;
    0 references
    public HomeController()
    {
        context = new ForestContext();
    }
```

**Implementing the *Index()* Action**

Objective; *Index()* action is to display the genres of music:
_ We need to implement the *Index()* Action.
_ We need to implement the view that will display the list of music.

_ Implement:

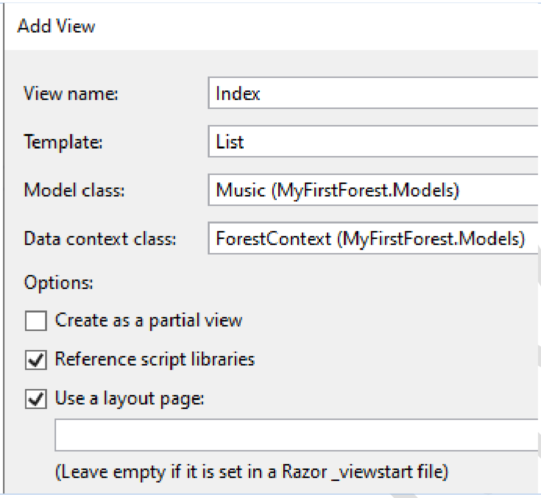


```
public ActionResult Index()
{
    IList<Music> musics = context.Music.ToList();
    return View(musics);
}
```

_ Within the *Solution Explorer*, right click on the project and *Build* the project.

Note: In creating the views, you would point a view to one of the available model classes within the project. In order to make the model classes available you need to compile (Build) them or build your solution. This is also to make sure that the solution does build and that we have thus far no errors. Generally, you should build your solution often. This helps you isolate errors that you can deal with in timely fashion.

_ Anywhere within the *index* action, right click and *Add View*.



Add View

View name: Index

Template: List

Model class: Music (MyFirstForest.Models)

Data context class: ForestContext (MyFirstForest.Models)

Options:

☐ Create as a partial view

☑ Reference script libraries

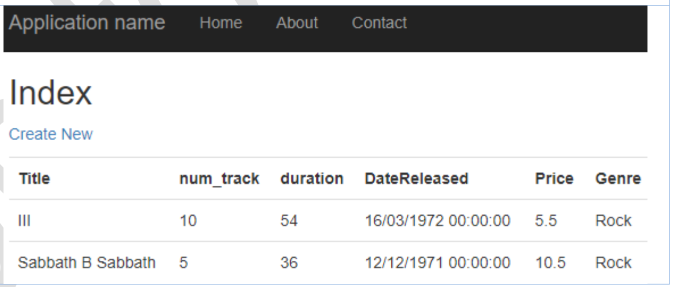
☑ Use a layout page:

(Leave empty if it is set in a Razor _viewstart file)

Inspect the *View* code: Note the model property at the top of the view file:
What is the model? What is the relation of model to the View code?

_ Build and run the application.

To run, press F5
Or
DEBUG – Start Debugging
Or
Click the play button ( ▶ )



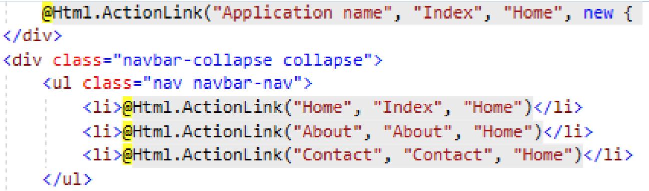
Index

Create New

| Title | num_track | duration | DateReleased | Price | Genre |
|---|---|---|---|---|---|
| III | 10 | 54 | 16/03/1972 00:00:00 | 5.5 | Rock |
| Sabbath B Sabbath | 5 | 36 | 12/12/1971 00:00:00 | 10.5 | Rock |

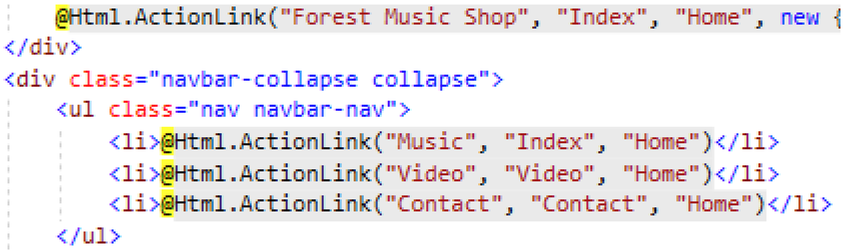
**Customising the application**
*Forest* is an online shop for music and video. Currently we have links on the page such as Home, About, Contact. Aims is to customise some of these links.

_ Open the file Views/Shared/_Layout.cshtml and at about line number 20 find as shown and alter



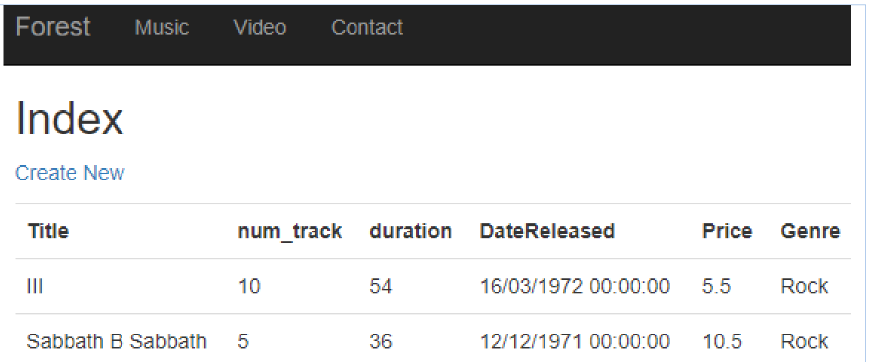
```
        @Html.ActionLink("Application name", "Index", "Home", new {
    </div>
    <div class="navbar-collapse collapse">
        <ul class="nav navbar-nav">
            <li>@Html.ActionLink("Home", "Index", "Home")</li>
            <li>@Html.ActionLink("About", "About", "Home")</li>
            <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
        </ul>
```

To



```
    @Html.ActionLink("Forest Music Shop", "Index", "Home", new {
</div>
<div class="navbar-collapse collapse">
    <ul class="nav navbar-nav">
        <li>@Html.ActionLink("Music", "Index", "Home")</li>
        <li>@Html.ActionLink("Video", "Video", "Home")</li>
        <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
    </ul>
```

_Save and run the application
_ Inspect the links



Forest Music Video Contact

Index

Create New

| Title | num_track | duration | DateReleased | Price | Genre |
|---|---|---|---|---|---|
| III | 10 | 54 | 16/03/1972 00:00:00 | 5.5 | Rock |
| Sabbath B Sabbath | 5 | 36 | 12/12/1971 00:00:00 | 10.5 | Rock |

Click on the links and inspect where they take you.
Reflect on how the links work.