****

**Migration in code first approach**

**Date: 23/05/18**

**Name: Amarjit Kumar( M1044368), Shivani Shinha**

**Project: DriveSafe**

**Track: DotNet**

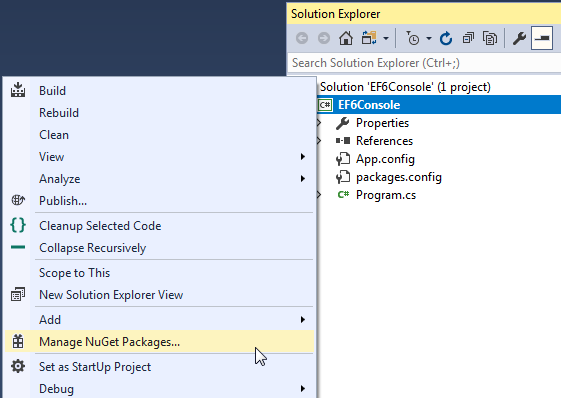
**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Introduction:**

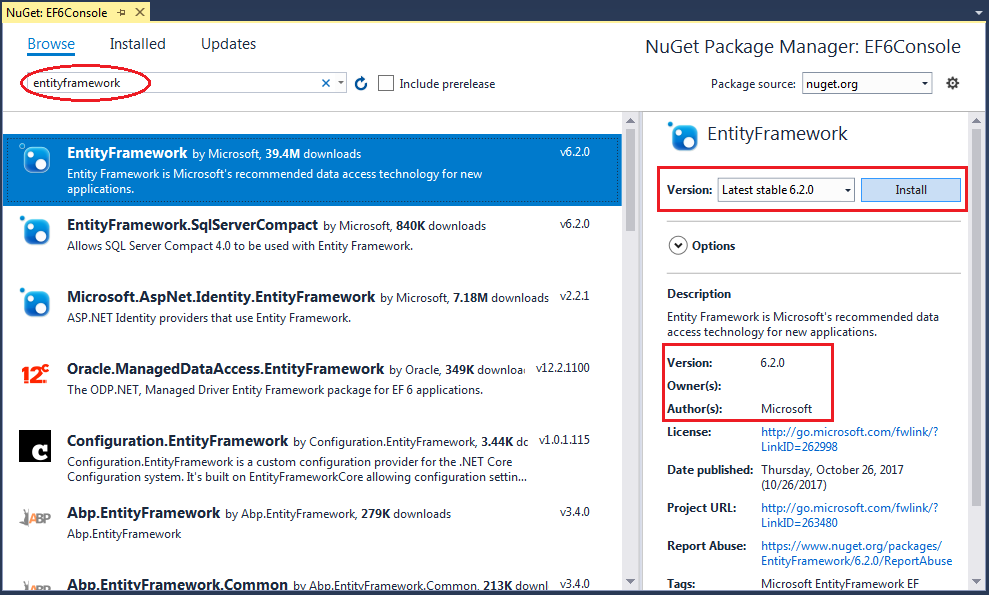
Entity Framework introduced automated migration so that you don't have to process database migration manually for each change you make in your domain classes. Using migration we can easily create our new database with changed schema.

**How to do?**

First of all we have to install entity framework for our project.

****

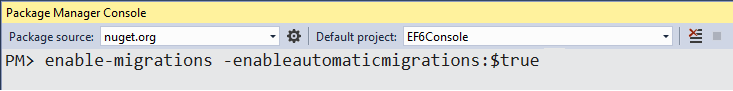
Right click on your project source file and select Manage NuGet Packages.

****

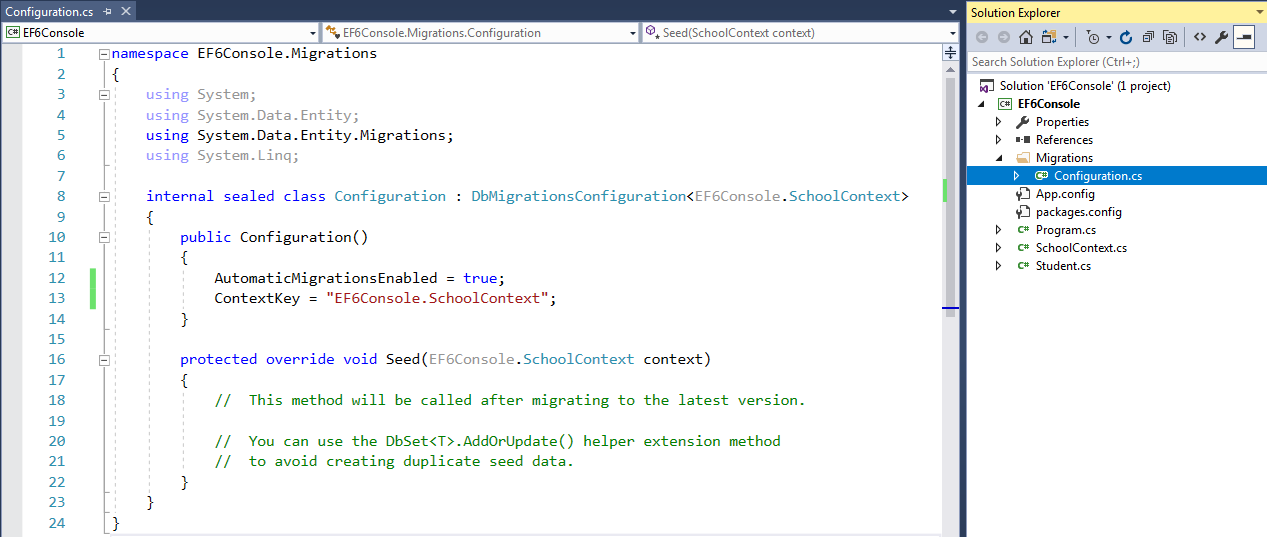
After that open NuGet package manager console.

How to do that?

The automated migrations can be implemented by executing the enable-migrations command in the Package Manager Console. Open the Package Manager Console from **Tools → Library Package Manager → Package Manager Console** and then run the enable-migrations –EnableAutomaticMigration:$true command (make sure that the default project is the project where your context class is).



Once the command runs successfully, it creates an internal sealed Configuration class derived from DbMigrationConfiguration in the Migration folder in your project:



As you can see in the constructor of the Configuration class, AutomaticMigrationsEnabled is set to true.

The next step is to set the database initializer in the context class to MigrateDatabaseToLatestVersion, as shown below.

public class SchoolContext: DbContext

{

public SchoolDBContext(): base("SchoolDB")

{

Database.SetInitializer(new MigrateDatabaseToLatestVersion<SchoolDBContext, EF6Console.Migrations.Configuration>());

}

public DbSet<Student> Students { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)

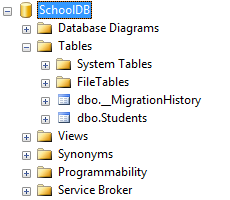
{

base.OnModelCreating(modelBuilder);

}

}

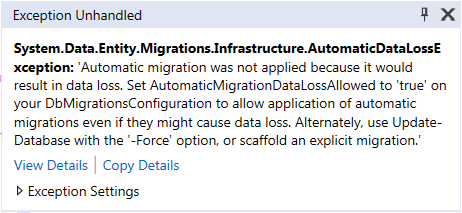
Now, you are all set for automated migration. EF will automatically take care of the migration when you change the domain classes. As of now, we only have the Student entity as per the SchoolContext class above. Run the application and look at the created database:

****

You will find that EF API created a system table \_\_MigrationHistory along with the Students table. The \_\_MigrationHistory table contains the history of database changes for all the migrations.

Now, you can add new domain classes and when you run the application again and you will see that the database contains tables for all entities automatically. You don't need to run any command.

However, this works only if you add new domain classes or remove classes, but it won't work when you add, modify or remove properties in the domain classes. To do this, remove any property from any domain class and run the application. You will get the following exception.

****

In your project, go to the Package Manager windows and:

1. Enable migrations: Enable-Migrations
2. Create migration: Add-Migration Initial
3. Create upgrade/downgrade script: Update-Database

In you case you are adding a new table (Color) and two new columns.

You have reason, if you start your website the FavoriteColor will be deleted including its data.

**What you can do:**

In the package manager console when you run Add-Migration Initial that will create a new script (C# file). As you can see in this file there is one column deleted, 2 added and 1 table created.

Make sure the table is created before the columns, populate it with data, create your 2 columns with existing data based on old column and then delete the column.

Another way (perhaps better) is to do this in several migration scripts, you can also use the Seed()method to populate data.

**Problems Faced:**

1. **Connection String could not found**
2. **Can not find the Entity framework**

First problem can be solved by right clicking on project file that contains configuration files, select ‘set up as startup Project’.

Second problem can be solved by selecting project in which entity framework is installed, in Package Source.