# Entity Framework Lab 1

Core Entity Framework Tutorial

This tutorial is taken from `https://docs.microsoft.com/en-us/ef/core/get-started/netcore/new-db-sqlite` and `https://github.com/aspnet/EntityFramework.Docs/tree/master/samples/core/GetStarted/NetCore`.

## 1 Beginning the lab

1. Create a new console project. `File` ▶ `New` ▶ `Project` ▶ `Console App C#` ▶ `Next`. See figure 1. Configure your project as shown in figure 2.

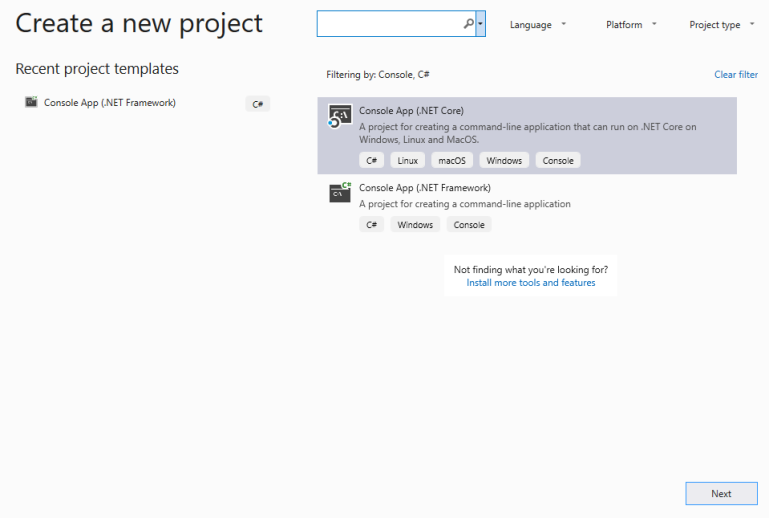


Figure 1: Beginning a new Console App

2. Go to `Tools` ▶ `NuGet PackageManage` ▶ `Manage Packages for Solution` and search for `Microsoft.EntityFrameworkCore.Sqlite`. Install the package, see figure 3. Also, install `Microsoft.EntityFrameworkCore.Design`.

3. Add a new class named `BloggingContext.cs`. Right click on the project and select `Add` ▶ `Class`. See figure 4. Edit the file as shown in listing 1.

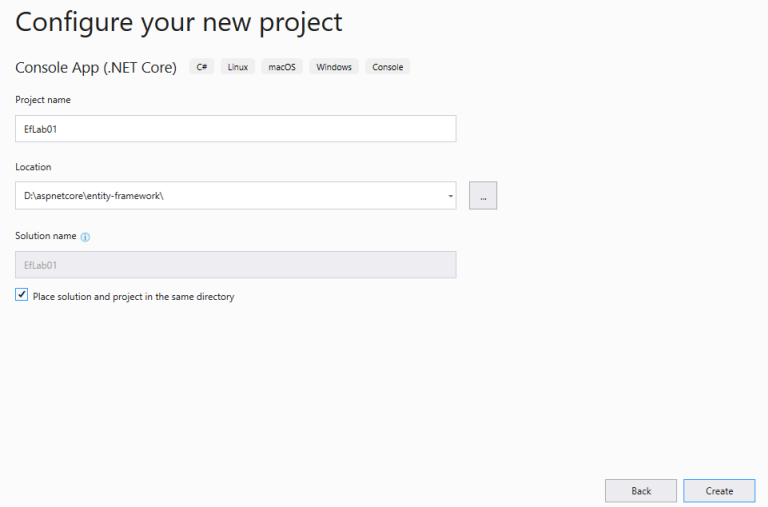


Figure 2: Configure a new Console App

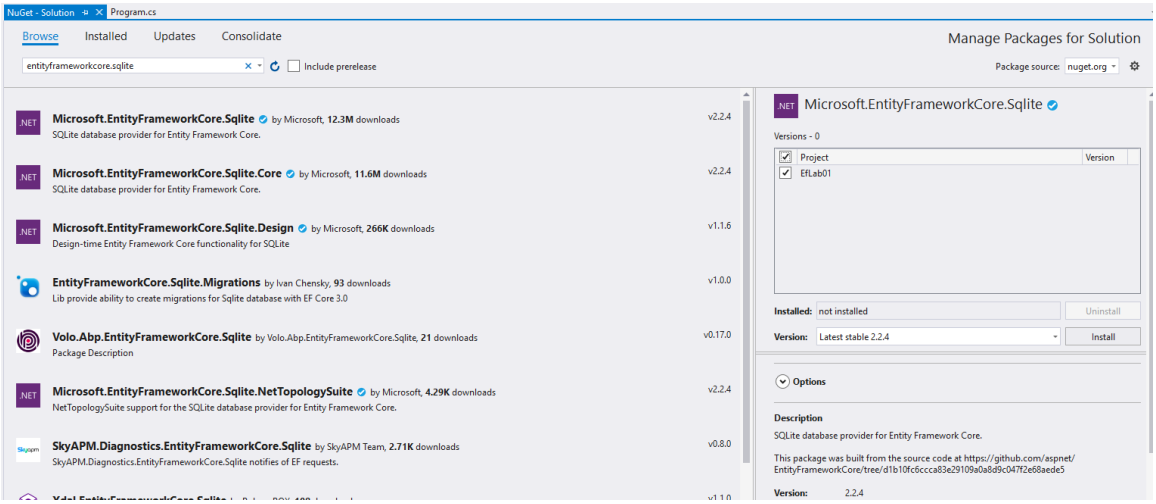


Figure 3: Install `Microsoft.EntityFrameworkCore.Sqlite`

Listing 1: Contents of `BloggingContext`

```
using Microsoft.EntityFrameworkCore;
using System.Collections.Generic;

namespace EfLab01
{
    public class BloggingContext : DbContext
    {
        public DbSet<Blog> Blogs { get; set; }
        public DbSet<Post> Posts { get; set; }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            optionsBuilder.UseSqlite("Data Source=blogging.db");
        }
    }
}
```

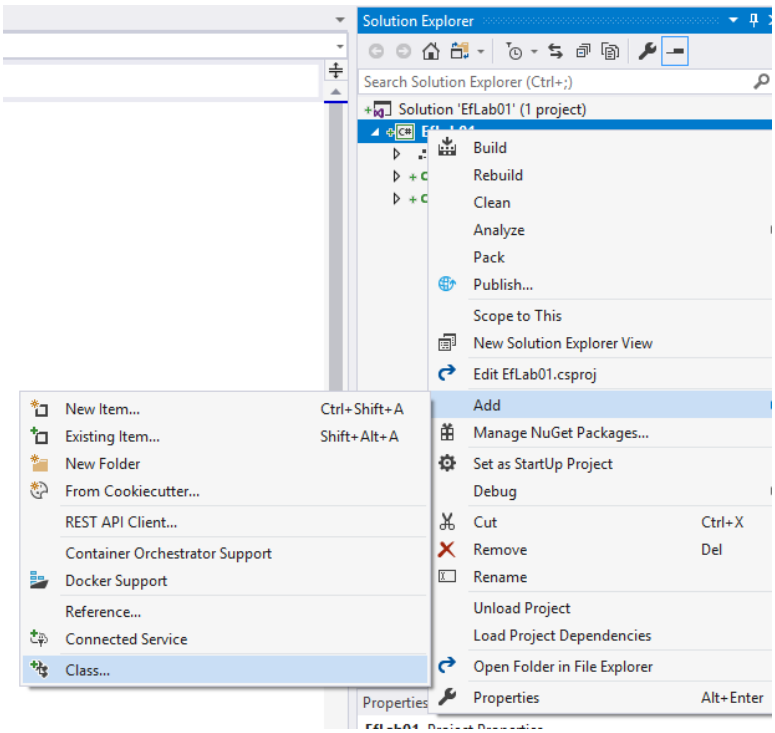


Figure 4: Create a new class named `BloggingCotext`

4. Add a new class named `Blog.cs`. Right click on the project and select `Add` ► `Class`.Edit the file as shown in listing 2.

Listing 2: Cotents of `Blog.cs`

```
using Microsoft.EntityFrameworkCore;
using System.Collections.Generic;

namespace EfLab01
{
    public class Blog
    {
        public int BlogId { get; set; }
        public string Url { get; set; }

        public ICollection<Post> Posts { get; set; }
    }
}
```

5. Add a new class named `Post.cs`. Right click on the project and select `Add` ► `Class`.Edit the file as shown in listing 3.

Listing 3: Contents of `Post.cs`

```
using Microsoft.EntityFrameworkCore;
using System.Collections.Generic;

namespace EfLab01
{
    public class Post
    {
        public int PostId { get; set; }
        public string Title { get; set; }
        public string Content { get; set; }

        public int BlogId { get; set; }
        public Blog Blog { get; set; }
    }

}
```

6. Open a PowerShell prompt and navigate to the direcetory that contains your `Program.cs` file. See figure 5. Then, run the commands shown in listing 4.

```
PS D:\aspnetcore\entity-framework\EfLab01> dir


    Directory: D:\aspnetcore\entity-framework\EfLab01


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         5/7/2019   9:11 AM                bin
d-----         5/7/2019   9:28 AM                obj
-a----         5/7/2019   9:44 AM            278 Blog.cs
-a----         5/7/2019   9:44 AM            436 BloggingContext.cs
-a----         5/7/2019   9:44 AM            474 EfLab01.csproj
-a----         5/7/2019   9:44 AM            352 Post.cs
-a----         5/7/2019   9:11 AM            189 Program.cs


PS D:\aspnetcore\entity-framework\EfLab01>
```

Figure 5: Open a PowerShell console

Listing 4: PowerShell commands

```
dotnet ef migrations add InitialCreate
dotnet ef database update
```

7. Edit the `Program.cs` file to match listing 5.

Listing 5: Contents of `Program.cs`

```
using EfLab01;
using System;

namespace ConsoleApp.SQLite
{
    public class Program
    {
        public static void Main()
        {
            using (var db = new BloggingContext())
            {
                db.Blogs.Add(new Blog { Url = "http://blogs.msdn.com/adonet" });
                var count = db.SaveChanges();
                Console.WriteLine("{0} records saved to database", count);

                Console.WriteLine();
                Console.WriteLine("All blogs in database:");
                foreach (var blog in db.Blogs)
                {
                    Console.WriteLine(" - {0}", blog.Url);
                }
            }
        }
    }
}
```

8. To run your application, enter the command shown in listing 6 in PowerShell.

Listing 6: PowerShell command to run the application

```
dotnet run
```