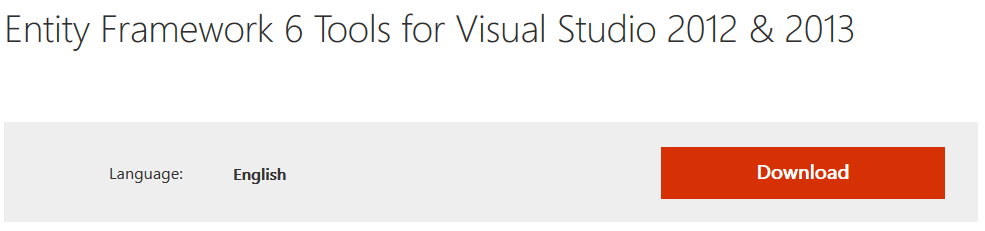
# Exercise: Working with Entity Framework

The purpose of this exercise is to make you **familiar** with the Entity Framework. After completing it, you should be able to **write and execute LINQ queries and Lambda expressions that manipulate your database**.

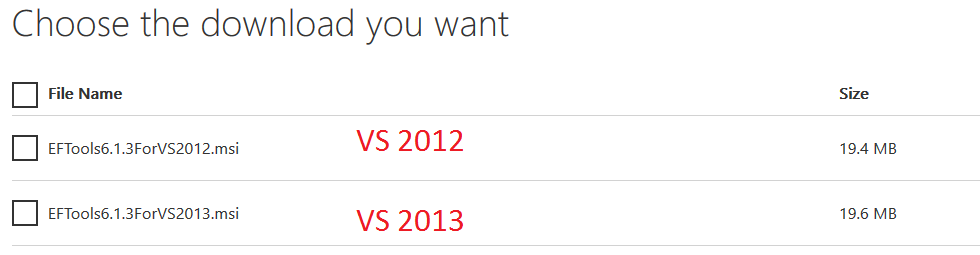
This lab is part of the [“Software Technologies” course @ SoftUni](https://softuni.bg/courses/software-technologies).

# Part I: Create Project and Add EF

**NOTE**: If you are using Visual Studio 2013 or 2012, you need to **install** ‘**Entity Framework 6 Tools**’. You can download them from this link - <https://www.microsoft.com/en-us/download/details.aspx?id=40762>.

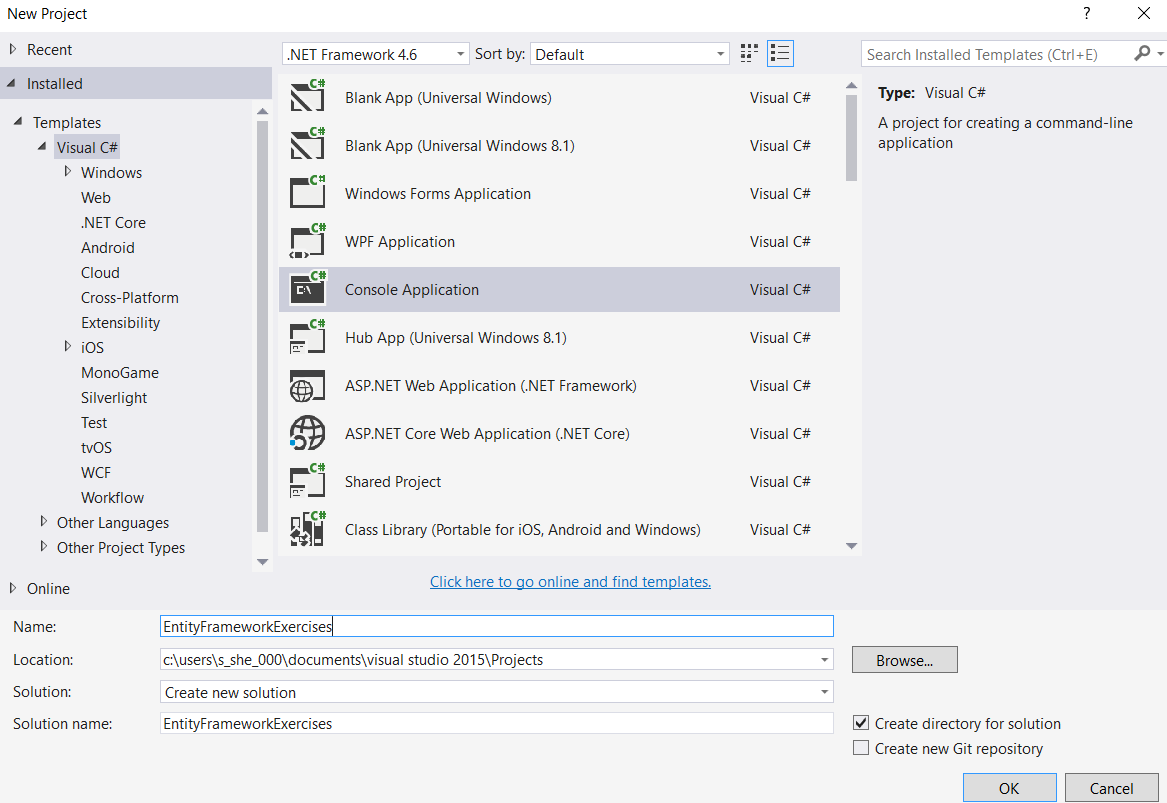


Clicking on download will lead you to this page:

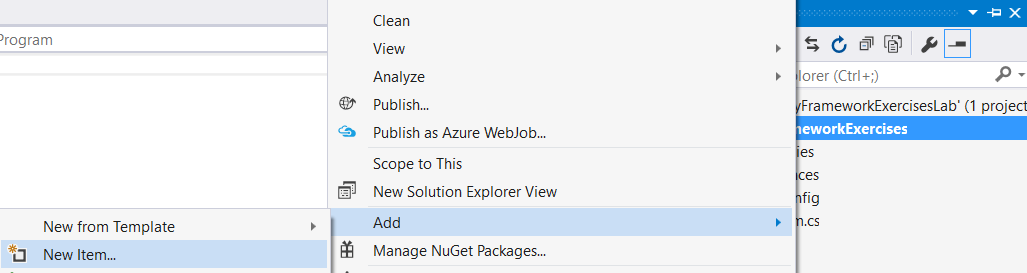


Before installing them **close** Visual Studio.

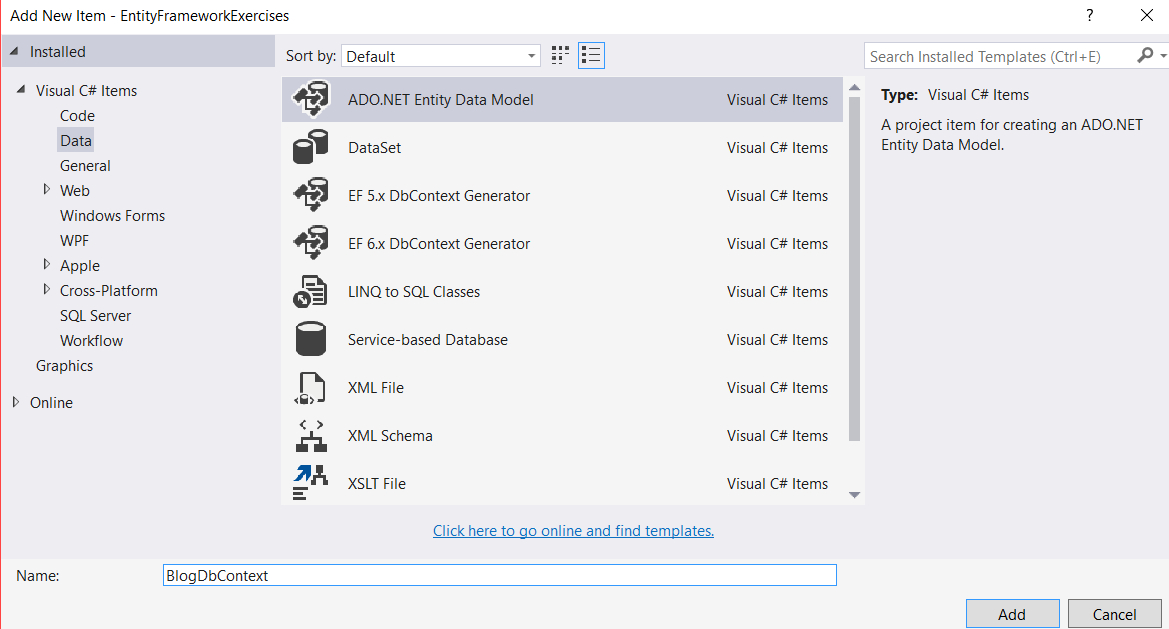
In Visual Studio, **create a new project**:



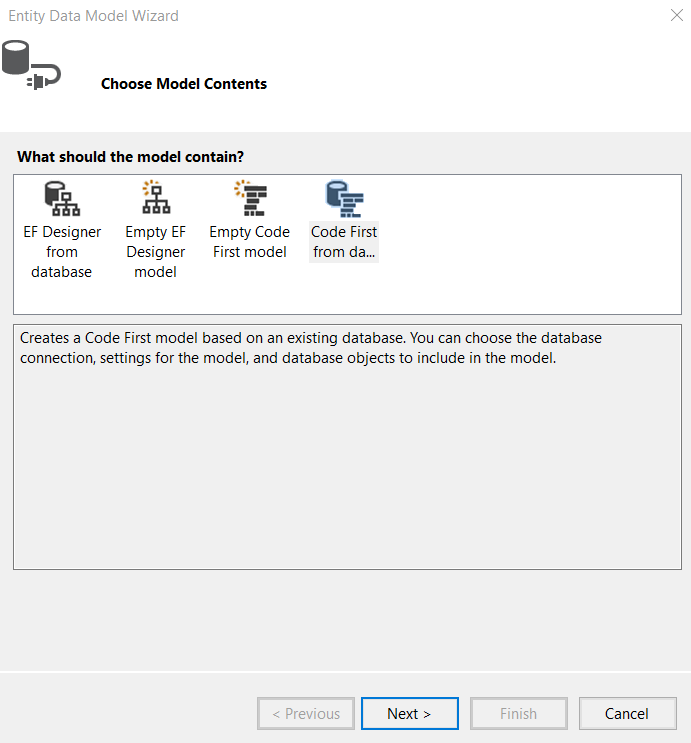
After the project is created right-click on it, and choose ‘**Add** -> **New** **Item’**:



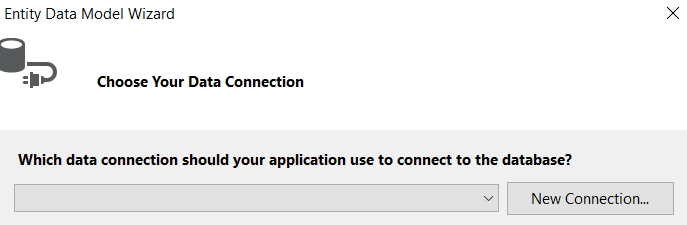
In the window that has just opened, click on **Data** from the list in the right side. Choose **ADO.NET Entity Data Model**.



Choose ‘**Code First from Database’**



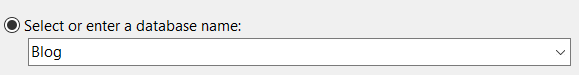
Now click on ‘**New Connection**’:



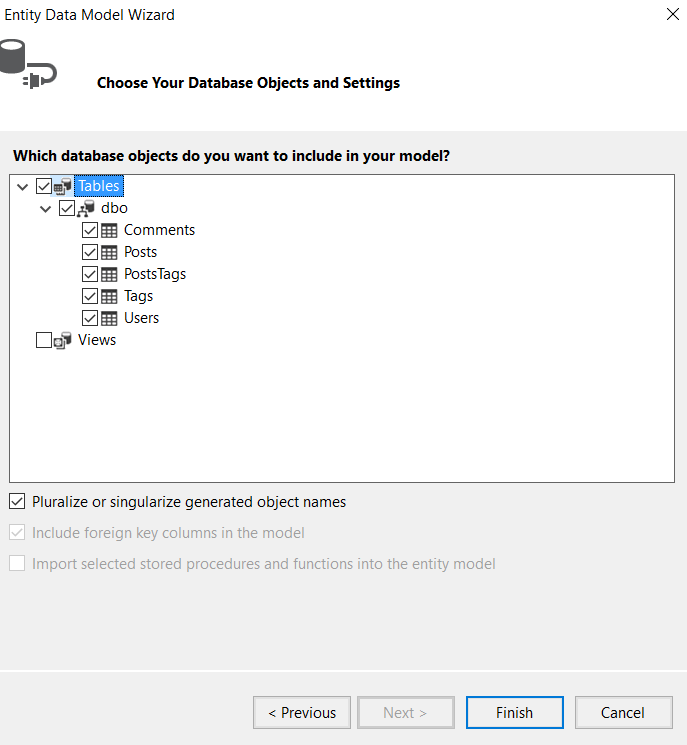
Enter the following data:



And choose the database:



Click ‘**OK**’ and then click ‘**Next**’. In the final page we need to select all tables from our database:



Now you are **ready to use Entity Framework**.

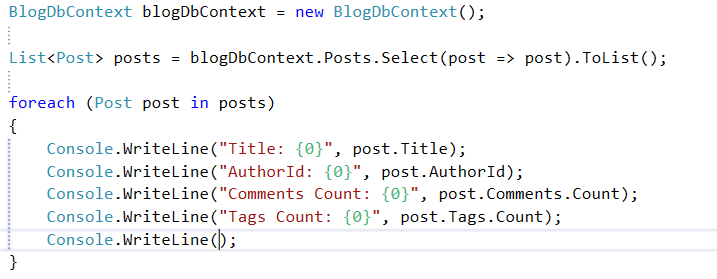
# Part II: Queries

## Read Data

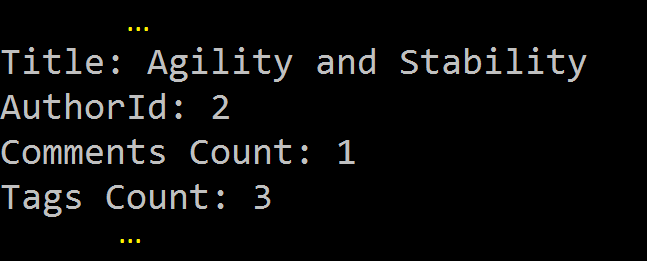
You should already be familiar with the Select() method. If you don’t remember how to use it, you can check the presentation for LINQ in the “[Programming Fundamentals](https://softuni.bg/trainings/1367/programming-fundamentals-may-2016)” course. It works similarly to the SELECT command in SQL.

### List All Posts

Let’s list **all data rows** from the Posts table.   
Go to your Main() method and write the following code:

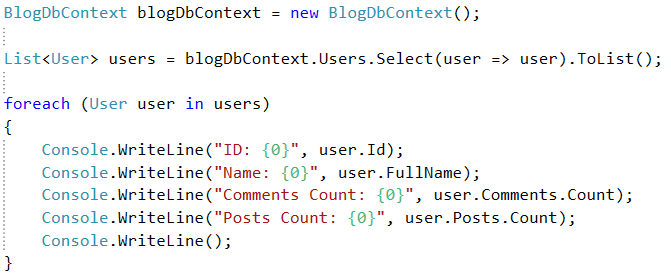


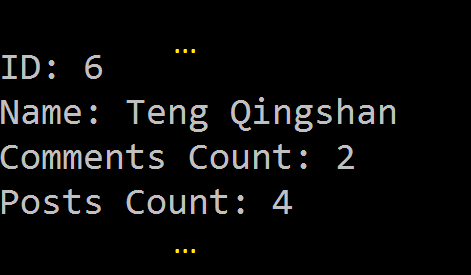
Run the code and you should see many posts like this one:



### List All Users

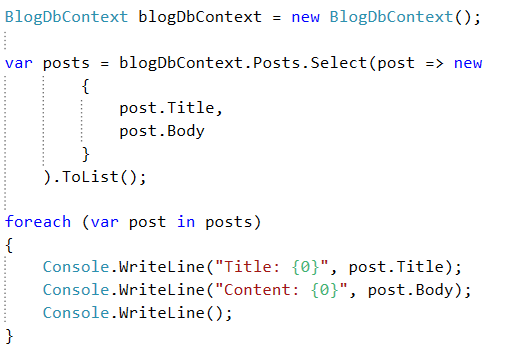
Let’s get **all users** now:

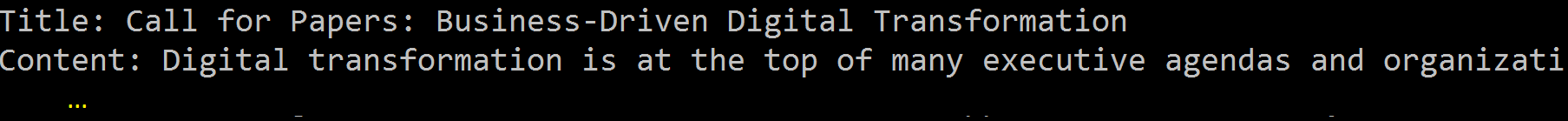




### List Title and Body from Posts

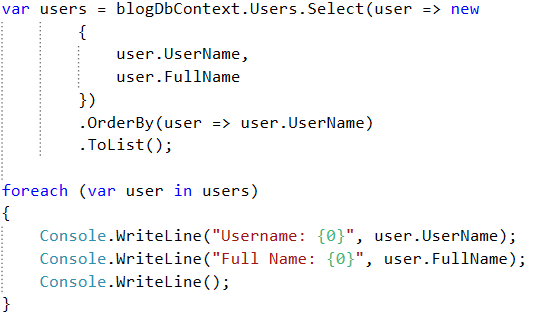
Let’s do something harder. We will get only the **title and body** from every post, ignoring the other fields:

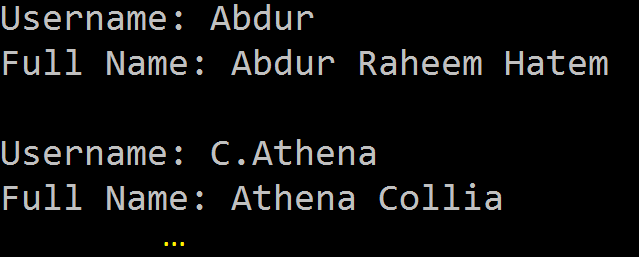




### Order Data

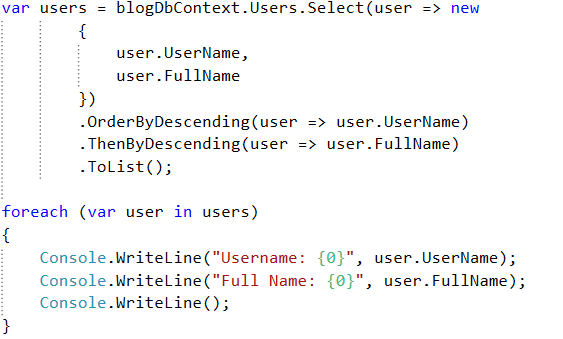
Let’s take all **usernames and full names** of our users. Now we will **order** the data in **ascending** order by **username**.

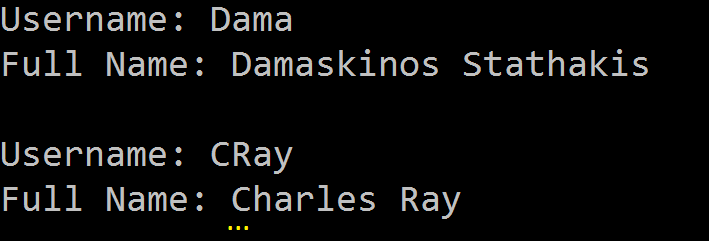




### Order by Two Columns

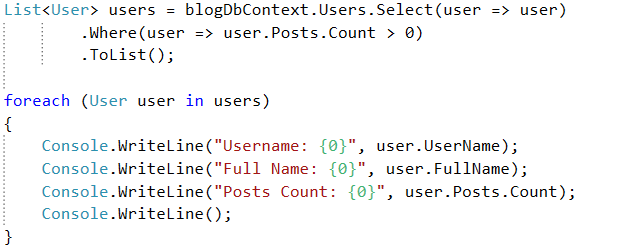
Let’s try to order by **2 parameters**, but this time we will do it in **descending** order:

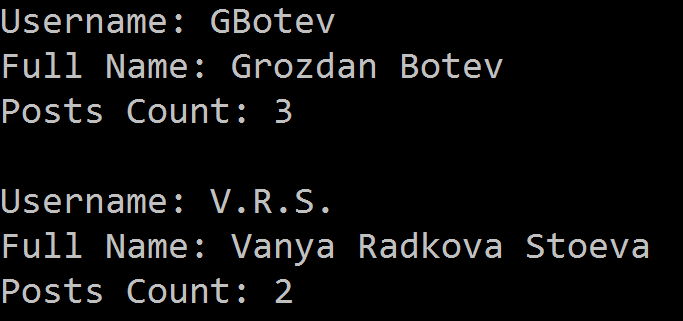




### Select Authors

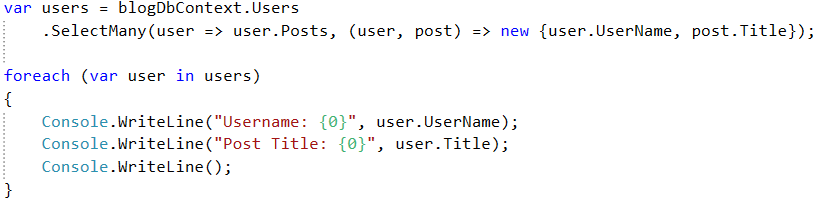
**Not all** of our users have **created posts**. How can we get **only the users that have created a blog post**? Just like that:

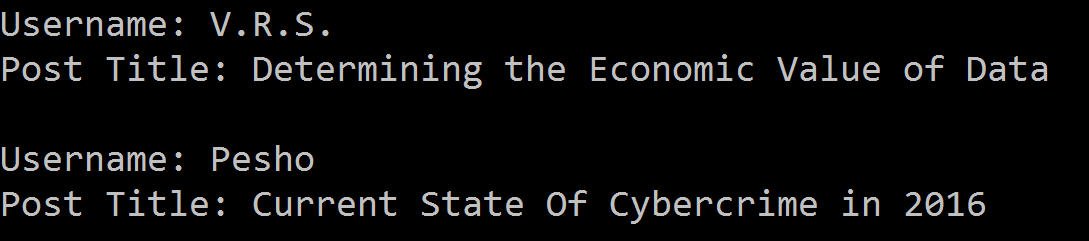




### Joins Authors with Titles

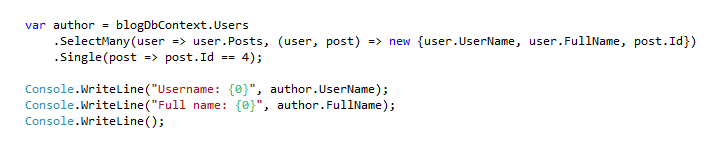
Something more interesting now – let’s get the usernames with the post names for each user. The query should look like this:

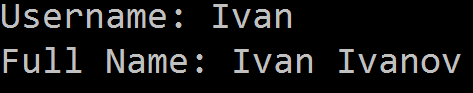




### Select Author of Specific Post

Let’s get the creator of post with id=4 now:





Another solution:

Console.WriteLine(db.Posts.Find(4).User.Username).

### Order Posts Authors

Let’s do one last Select() query. It should **get all of the post authors** and **order them** by id in **descending** order. We will need only the **username and full name**. But this time, you will be given only the expected result.



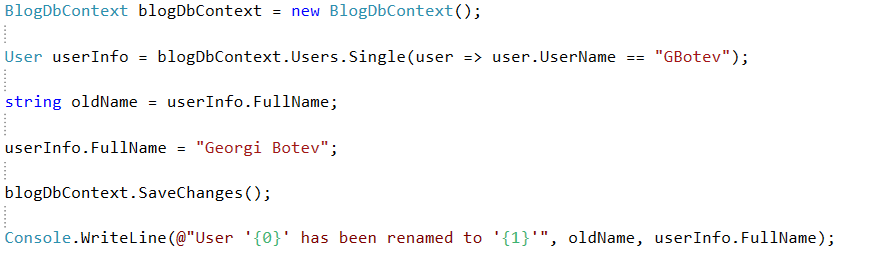
## Create Data

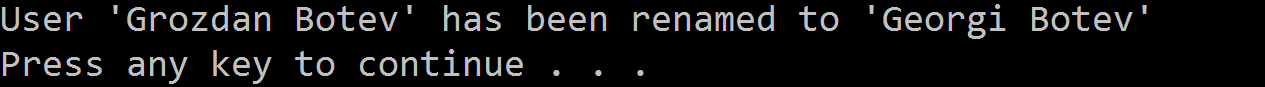
In order to **create** data, we need to **create object from the class** we desire to use. For this example, we are going to create a **new post**, which means that we need to create new **instance of** Post. Then we need to **save** our **database** with the new changes. Let’s try it!



## Update Data

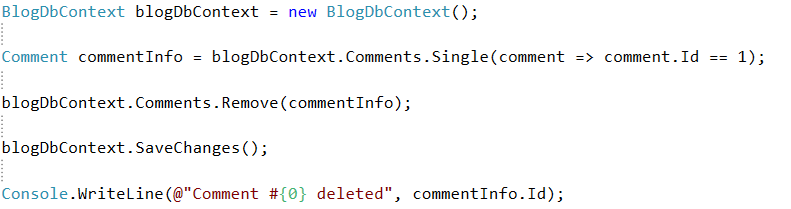
If we want to **update** an object, first we need to **get it from the database.** Then we can **edit** its properties, and **finally save the database**. This is how it’s done:





## Delete Data

In order to **delete** data from our tables, we need to **get the element** we want to delete. Let’s start with something simple – deleting comment from a post. We will the delete the comment with Id = 1.





Now let’s delete the post that we’ve created earlier. We will do that, **using the ID** of the post. After getting the object, we should **delete everything that is linked** to it, from the **other tables**. In our case, we should delete the **tags and comments**. Then we can simple use Remove() to delete the post.

