fogsoft

ASP.NET MVC: Entity Framework Code First

Яцканич Андрей

C#

ASP.NET MVC

Построение приложения

Цель

Познакомиться Entity Framework

- Подключение
- Использование
- Особенности генерации Code-First

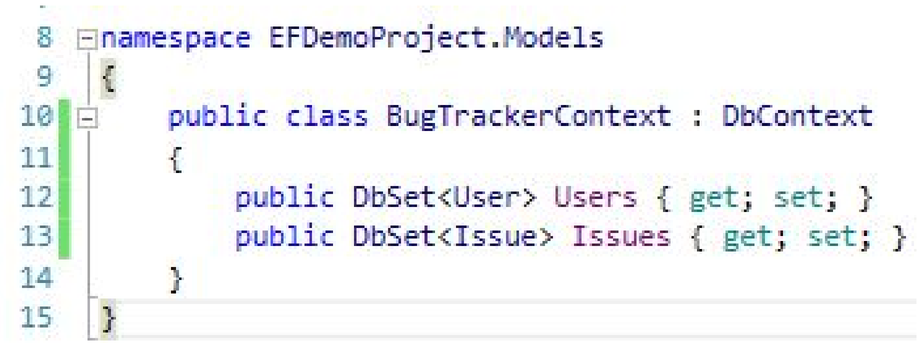
Связь приложения с базой данных

Entity Framework

Подходы в работе с EF

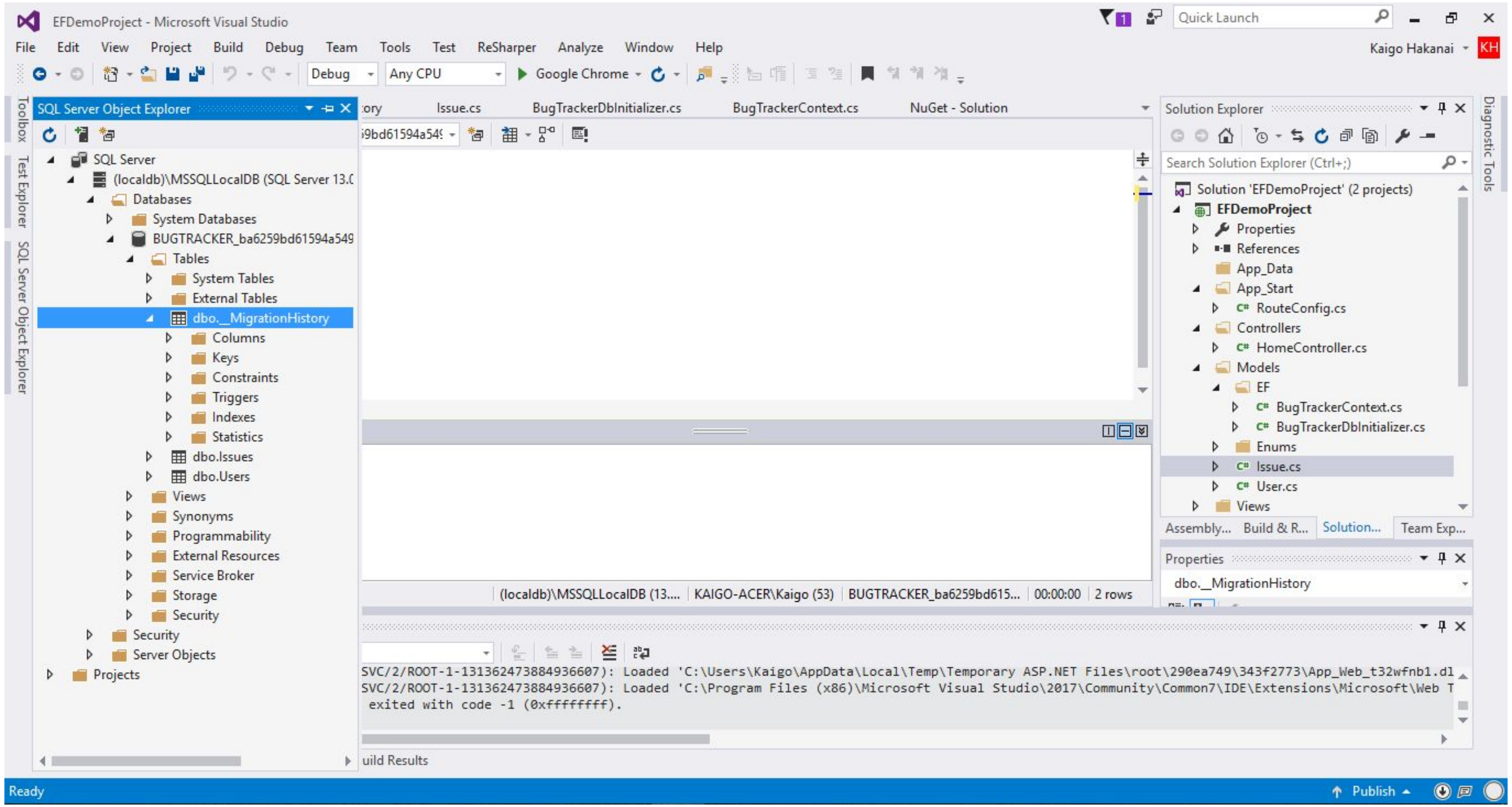
- Database-First
- Model-First
- Code-First

Создание и подключение базы данных



```
namespace EFDemoProject.Models
{
    public class BugTrackerContext : DbContext
    {
        public DbSet<User> Users { get; set; }
        public DbSet<Issue> Issues { get; set; }
    }
}
```

Изменение модели данных



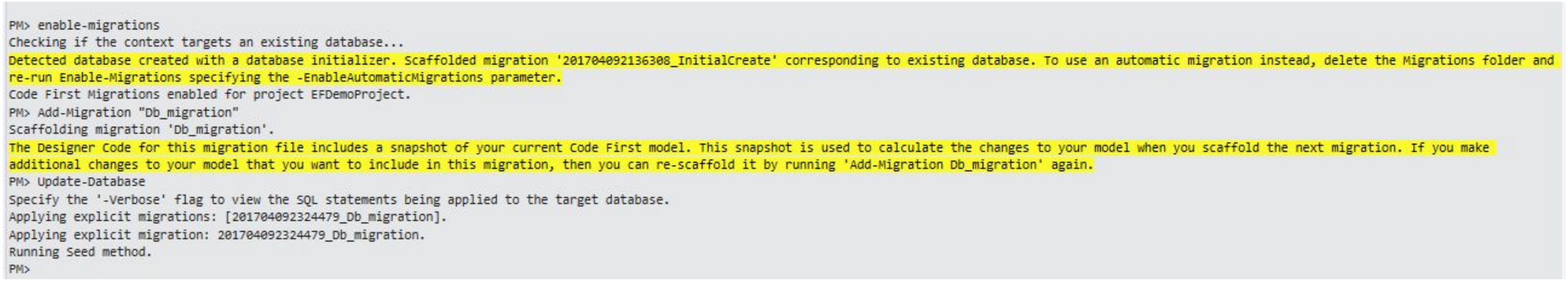
```
SVC/2/ROOT-1-131362473884936607): Loaded 'C:\Users\Kaigo\AppData\Local\Temp\Temporary ASP.NET Files\root\290ea749\343f2773\App_Web_t32wfnb1.dl
SVC/2/ROOT-1-131362473884936607): Loaded 'C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\Common7\IDE\Extensions\Microsoft\Web T
 exited with code -1 (0xffffffff).
```

Миграции базы данных

Enable-migrations

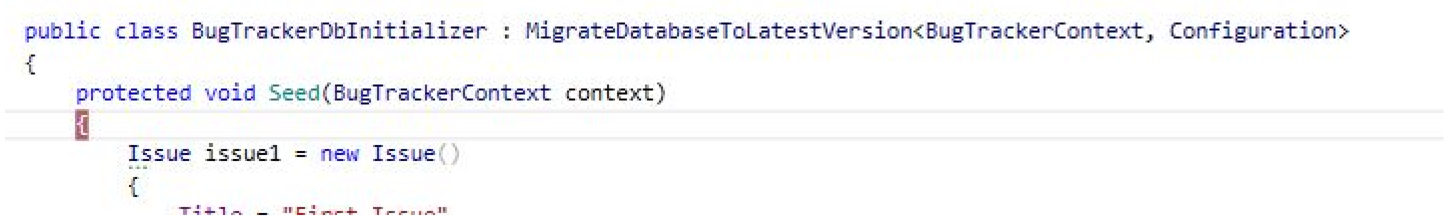
Add-Migration

Update-Database

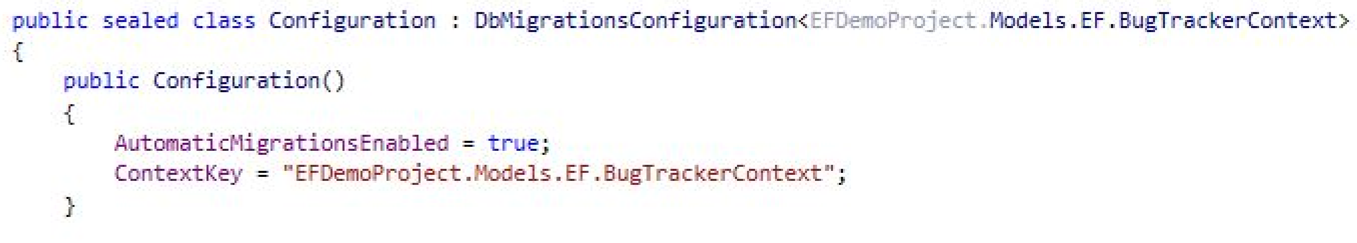


```
PM> enable-migrations
Checking if the context targets an existing database...
Detected database created with a database initializer. Scaffolded migration '201704092136308_InitialCreate' corresponding to existing database. To use an automatic migration instead, delete the Migrations folder and
re-run Enable-Migrations specifying the -EnableAutomaticMigrations parameter.
Code First Migrations enabled for project EFDemoProject.
PM> Add-Migration "Db_migration"
Scaffolding migration 'Db_migration'.
The Designer Code for this migration file includes a snapshot of your current Code First model. This snapshot is used to calculate the changes to your model when you scaffold the next migration. If you make
additional changes to your model that you want to include in this migration, then you can re-scaffold it by running 'Add-Migration Db_migration' again.
PM> Update-Database
Specify the '-Verbose' flag to view the SQL statements being applied to the target database.
Applying explicit migrations: [201704092324479_Db_migration].
Applying explicit migration: 201704092324479_Db_migration.
Running Seed method.
PM>
```

Автоматическая миграция



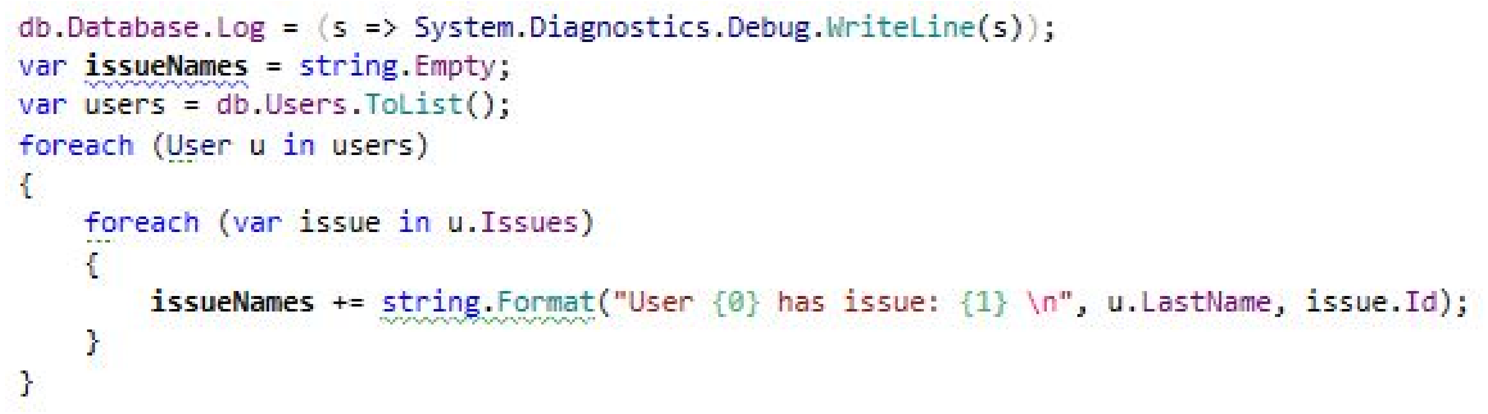
```
public class BugTrackerDbInitializer : MigrateDatabaseToLatestVersion<BugTrackerContext, Configuration>
{
    protected void Seed(BugTrackerContext context)
    {
        Issue issue1 = new Issue()
        {
            Title = "First Issue"
```



```
public sealed class Configuration : DbMigrationsConfiguration<EFDemoProject.Models.EF.BugTrackerContext>
{
    public Configuration()
    {
        AutomaticMigrationsEnabled = true;
        ContextKey = "EFDemoProject.Models.EF.BugTrackerContext";
    }
```

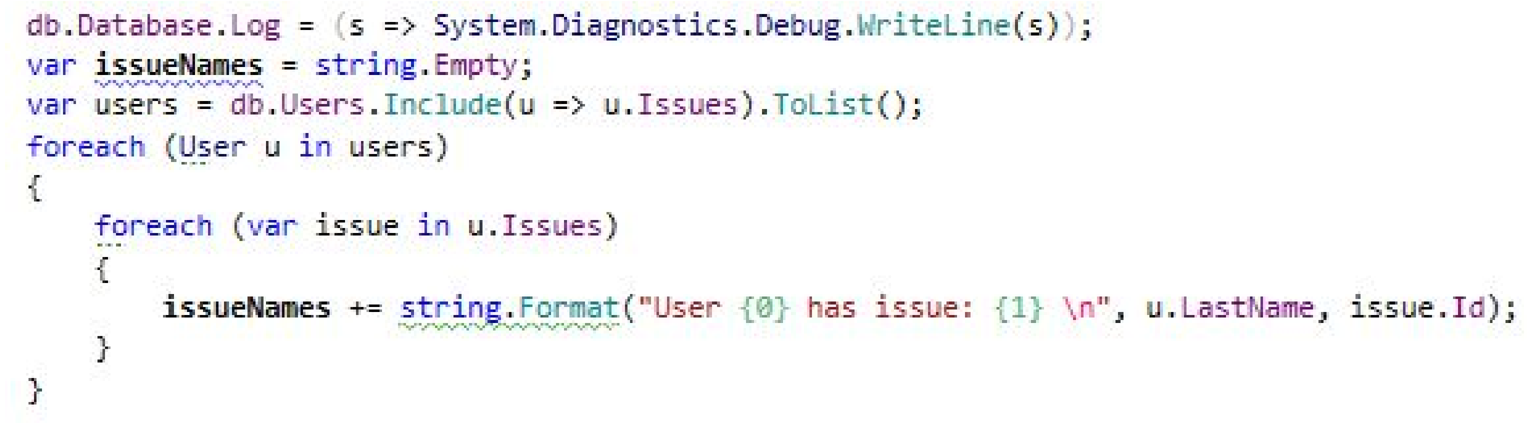
Загрузка данных EF

Lazyloading (Отложенная загрузка)



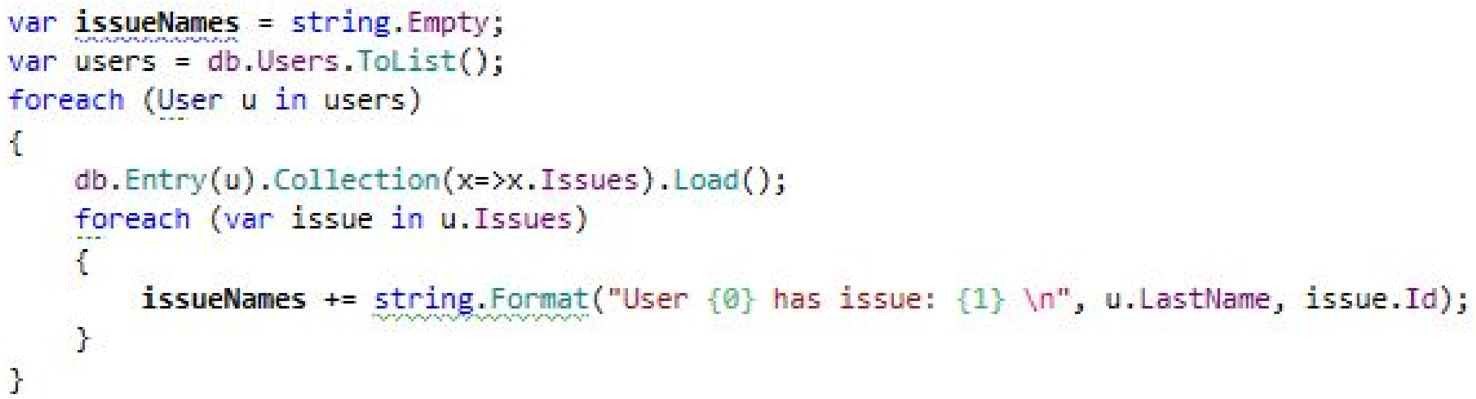
```
db.Database.Log = (s => System.Diagnostics.Debug.WriteLine(s));
var issueNames = string.Empty;
var users = db.Users.ToList();
foreach (User u in users)
{
    foreach (var issue in u.Issues)
    {
        issueNames += string.Format("User {0} has issue: {1} \n", u.LastName, issue.Id);
    }
}
```

Eagerloading (Прямая загрузка)



```
db.Database.Log = (s => System.Diagnostics.Debug.WriteLine(s));
var issueNames = string.Empty;
var users = db.Users.Include(u => u.Issues).ToList();
foreach (User u in users)
{
    foreach (var issue in u.Issues)
    {
        issueNames += string.Format("User {0} has issue: {1} \n", u.LastName, issue.Id);
    }
}
```

Explicitloading (Явная загрузка)

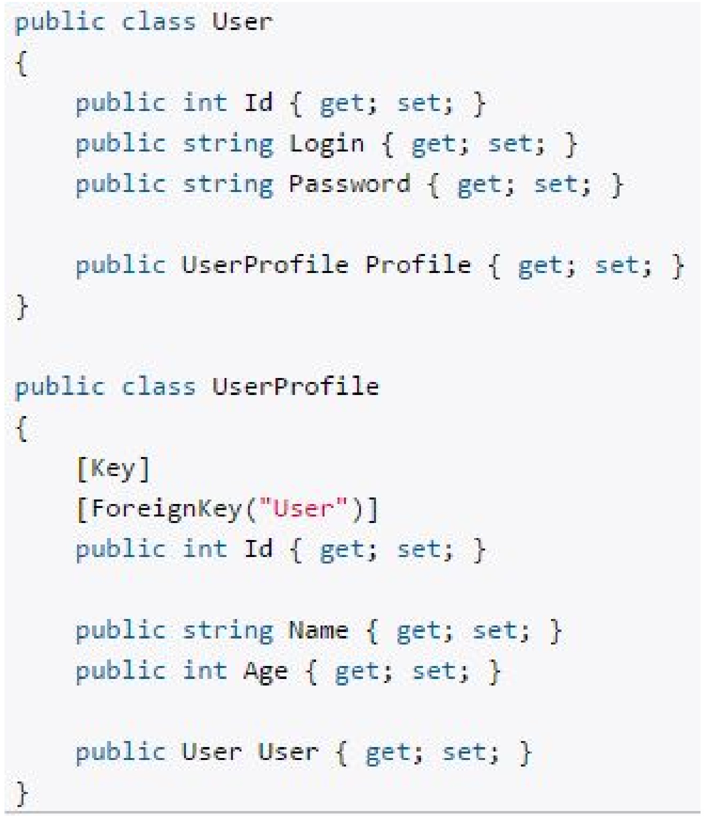


```
var issueNames = string.Empty;
var users = db.Users.ToList();
foreach (User u in users)
{
    db.Entry(u).Collection(x=>x.Issues).Load();
    foreach (var issue in u.Issues)
    {
        issueNames += string.Format("User {0} has issue: {1} \n", u.LastName, issue.Id);
    }
}
```

Связь данных

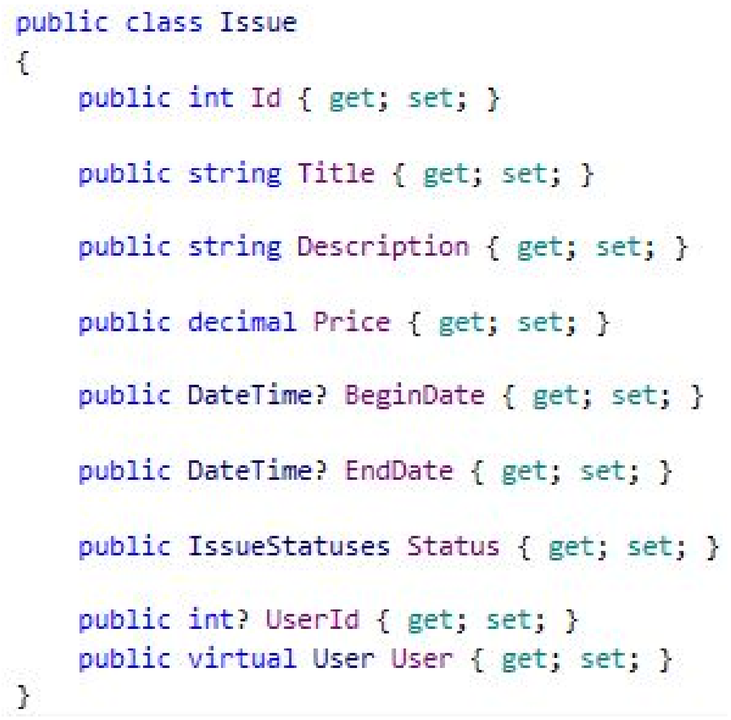
- one-to-one
- one-to-many
- Many-to-many

One-to-one



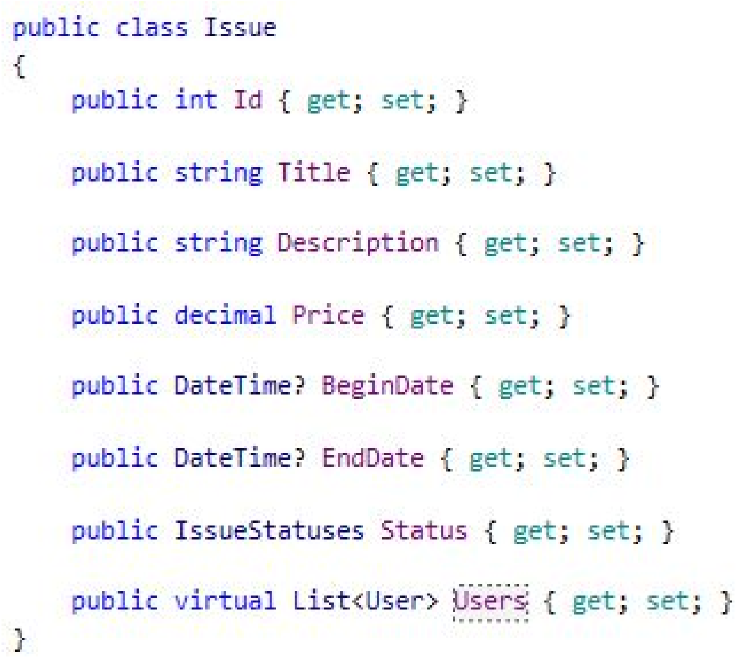
```
public class User
{
    public int Id { get; set; }
    public string Login { get; set; }
    public string Password { get; set; }

    public UserProfile Profile { get; set; }
}

public class UserProfile
{
    [Key]
    [ForeignKey("User")]
    public int Id { get; set; }

    public string Name { get; set; }
    public int Age { get; set; }

    public User User { get; set; }
}
```

One-to-many

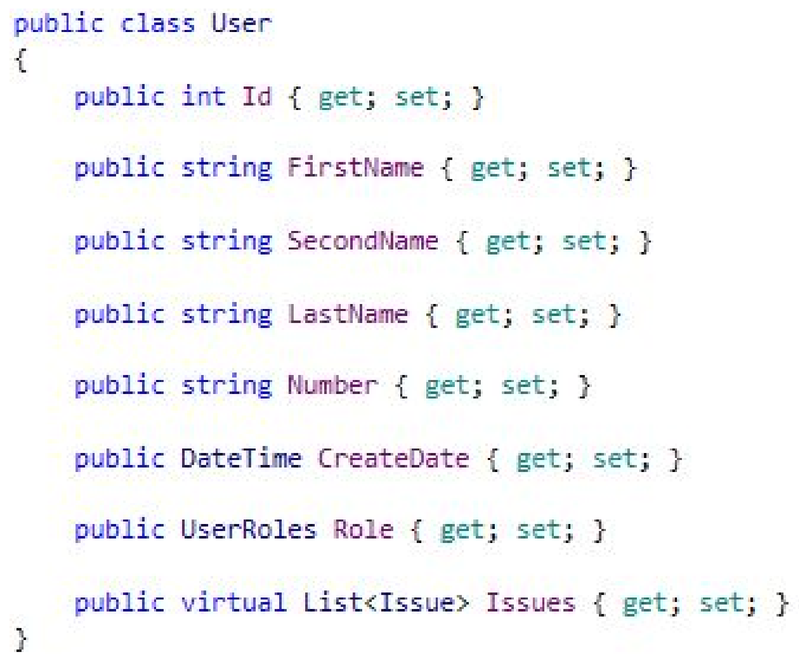


```
public class Issue
{
    public int Id { get; set; }

    public string Title { get; set; }

    public string Description { get; set; }

    public decimal Price { get; set; }

    public DateTime? BeginDate { get; set; }

    public DateTime? EndDate { get; set; }

    public IssueStatuses Status { get; set; }

    public int? UserId { get; set; }
    public virtual User User { get; set; }
}
```

Many-to-many



```
public class Issue
{
    public int Id { get; set; }

    public string Title { get; set; }

    public string Description { get; set; }

    public decimal Price { get; set; }

    public DateTime? BeginDate { get; set; }

    public DateTime? EndDate { get; set; }

    public IssueStatuses Status { get; set; }

    public virtual List<User> Users { get; set; }
}
```



```
public class User
{
    public int Id { get; set; }

    public string FirstName { get; set; }

    public string SecondName { get; set; }

    public string LastName { get; set; }

    public string Number { get; set; }

    public DateTime CreateDate { get; set; }

    public UserRoles Role { get; set; }

    public virtual List<Issue> Issues { get; set; }
}
```

Наши вакансии на сайте fogsoft.ru

Ждем ваших резюме!