PROPERTY CONFIGURATION

**WE WILL CONFIGURE EMPLOYEE CLASS**

**CHANGE MYDBCONTEXT**

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.ComponentModel.DataAnnotations.Schema;

namespace FluentAPI

{

class MyDbContext:DbContext

{

public MyDbContext():base("name:fluent")

{

}

public virtual DbSet<Department> Departments { get; set; }

public virtual DbSet<Employee> Employees { get; set; }

public virtual DbSet<EmployeeAddress> EmployeeAddresses { get; set; }

public DbSet<Project> Projects { get; set; }

public DbSet<Team> Teams { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

//Totable -> configure/set table name in sql server

modelBuilder.Entity<Department>().ToTable("Dept");

//HasKey (cause usually primary key will be either

//ID of DepartmentId in the class if it isnt so then we

//have to explicitly mention here

modelBuilder.Entity<Department>().HasKey(d=>d.DeptId);

//Property configuration "Employee" entity

// we dont want auto increment - >DatabaseGeneratedOption.None

modelBuilder.Entity<Employee>().Property(e=>e.EmployeeId).HasDatabaseGeneratedOption(DatabaseGeneratedOption.None);

//EmpNAme - change name to Ename

modelBuilder.Entity<Employee>().Property(e => e.EmpName).HasColumnName("Ename");

//EmpNAme - Not null

modelBuilder.Entity<Employee>().Property(e => e.EmpName).IsRequired();

}

}

}

Can be changed

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.ComponentModel.DataAnnotations.Schema;

namespace FluentAPI

{

class MyDbContext:DbContext

{

public MyDbContext():base("name:fluent")

{

}

public virtual DbSet<Department> Departments { get; set; }

public virtual DbSet<Employee> Employees { get; set; }

public virtual DbSet<EmployeeAddress> EmployeeAddresses { get; set; }

public DbSet<Project> Projects { get; set; }

public DbSet<Team> Teams { get; set; }

Method chaining instead of writing again and again

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

//Totable -> configure/set table name in sql server

modelBuilder.Entity<Department>().ToTable("Dept");

//HasKey (cause usually primary key will be either

//ID of DepartmentId in the class if it isnt so then we

//have to explicitly mention here

modelBuilder.Entity<Department>().HasKey(d=>d.DeptId);

//Property configuration "Employee" entity

// we dont want auto increment - >DatabaseGeneratedOption.None

modelBuilder.Entity<Employee>().Property(e=>e.EmployeeId).HasDatabaseGeneratedOption(DatabaseGeneratedOption.None);

//EmpNAme - change name to Ename

modelBuilder.Entity<Employee>().Property(e => e.EmpName).HasColumnName("Ename")

.IsRequired().HasMaxLength(50).HasColumnName("varchar");

}

}

}

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.ComponentModel.DataAnnotations.Schema;

namespace FluentAPI

{

class MyDbContext:DbContext

{

public MyDbContext():base("name:fluent")

{

}

public virtual DbSet<Department> Departments { get; set; }

public virtual DbSet<Employee> Employees { get; set; }

public virtual DbSet<EmployeeAddress> EmployeeAddresses { get; set; }

public DbSet<Project> Projects { get; set; }

public DbSet<Team> Teams { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

//Totable -> configure/set table name in sql server

modelBuilder.Entity<Department>().ToTable("Dept");

//HasKey (cause usually primary key will be either

//ID of DepartmentId in the class if it isnt so then we

//have to explicitly mention here

modelBuilder.Entity<Department>().HasKey(d=>d.DeptId);

//Property configuration "Employee" entity

// we dont want auto increment - >DatabaseGeneratedOption.None

modelBuilder.Entity<Employee>().Property(e=>e.EmployeeId).HasDatabaseGeneratedOption(DatabaseGeneratedOption.None);

//EmpNAme - change name to Ename

modelBuilder.Entity<Employee>().Property(e => e.EmpName).HasColumnName("Ename")

.IsRequired()

.HasMaxLength(50)

.HasColumnName("varchar");

//Primary Contact

modelBuilder.Entity<Employee>().Property(e => e.PrimaryContact).IsRequired()

.HasMaxLength(10);

}

}

}