بسم اللّه الرحمن الرحیم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

استراتژی های راه اندازی پایگاه داده درCode-First

مدرس: مهندس افشین رفوآ

آدرس آموزشگاه. تهران - خیابان شریعتی - بالا تر از خیابان ملک - جنب بانک صادرات - پلاک 561 - واحد 7
88146323 - 88446780 - 88146330

استراتژی های راه اندازی پایگاه داده درCode-First

شما در حال حاضر پایگاه داده یتان را بعد از اجرای code first application در همان مرحله اول ایجاد کرده اید اما هنگام اجرا برای دوم چگونه این کار انجام خواهد شد؟ آیا هر بار که شما Application را اجرا می کنید یک پایگاه داده جدید ایجاد می کند؟ در مورد محیط تولید چطور؟ چگونه پایگاه داده را تغییر می دهید زمانی که شما مدل Domain تان را تغییییر می دهید؟ برای اجرای این سناریوها شما باید از یکی از استراتژی های Database Initializer استفاده کنید.

چهار استراتژی متفاوت Database Initializer وجود دارند.

CreateDatabaseIfNotExists : این حالت پیش فرض است، همان طور که از اسمش پیداست پایگاه داده را زمانی که طبق پیکربندی هیچ پایگاه داده ای وجود نداشته باشد را ایجاد می کند. هر چند اگر شما مدل کلاس را تغییر داده و سپس application را با این Initializer اجرا کنید، خوب یک خطا یا Exception روی خواهد داد.

DropCreateDatabaseIfModelChanges : اگر مدل کلاس هایتان (entity classes) تغییر کنند، این Database Initializer موجود را حذف نموده و یک پایگاه داده جدید را ایجاد می کند. بنابراین موقعی که مدل کلاس هایتان تغییر کند نگران حفظ الگو (Schema) پایگاه داده یتان نباشید.

DropCreateDatabaseAlways : همان طور که از اسمش پیداست این Initializer هر دفعه که شما Application را اجرا کنید پایگاه داده موجود را حذف می کند، صرف نظر از اینکه مدل کلاس هایتان تغییر کرده یا نه. این می تواند زمانی مفید باشد که شما در هر دفعه از اجرای Application پایگاه داده جدید بخواهید و همین طور در زمان توسعه Application.

Custom DB Initializer : اگر هیچ یک از راه اندازهای ایجاده پایگاه داده در بالا نیاز های شما را تامین نمی کند یا شما می خواهید فرایند دیگری را که پایگاه داده را با استفاده از initializer های بالا آغاز کنید، خوب در این صورت می توانید initializer سفارشی خودتان را ایجاد کنید.

برای استفاده از استراتژی های DB initialization مطرح شده در بالا، باید DB Initializer را با استفاده از کلاس پایگاه داده در کلاس Context به شرح زیر به کار برید.

```
      public class SchoolDBContext: DbContext
            {
                public SchoolDBContext():
base("SchoolDBConnectionString")
                {
                    Database. SetInitializer<schooldbcontext>
            (new CreateDatabaseIfNotExists<schooldbcontext>
                ());
                        //Database. SetInitializer
                    (new
DropCreateDatabaseIfModelChanges<schooldbcontext>
              ());
                              //Database. SetInitializer
                              (new
DropCreateDatabaseAlways<schooldbcontext>
                                  ());
                                    //Database. SetInitializer
                            (new SchoolDBInitializer());
                }
                                  public DbSet<student>
                                            Students { get; set; }
                  public DbSet<standard>  Standards { get; set;
}
            }
        </standard>
                                        </student>
                                    </schooldbcontext>
                                </schooldbcontext>
                            </schooldbcontext>
                        </schooldbcontext>
                    </schooldbcontext>
                </schooldbcontext>
```

شما می توانید DB initializer خودتان را به وسیله ارث بری از یکی از initializer ها همان طور که در پایین نشان داده شده است راایجاد کنید.

```
  public class SchoolDBInitializer :  DropCreateDatabaseAlways
            {
                protected override void Seed(SchoolDBContext
context)
                {
                    base. Seed(context);
                }
            }
```

همان طور که در کد بالا می بینید ما کلاس جدید SchoolDBInitializer را که از DropCreateDatabaseAlways مشتق گرفته شده است را ایجاد کرده ایم.

تنظیم db initializer در فایل configuration

شما می تواند db initializer در فایل configuration را نیز تنظیم کنید. برای مثال برای تنظیم پیش فرض initializer در app. config باید.

```
   <?xml version="1. 0" encoding="utf-8" ?>
           <configuration>
                 <appsettings>
                       <add key="DatabaseInitializerForType
SchoolDataLayer. SchoolDBContext, SchoolDataLayer" =
                                 = value="System. Data. Entity.
DropCreateDatabaseAlways`1[[SchoolDataLayer. SchoolDBContext,
SchoolDataLayer]], EntityFramework" />
           </appsettings>
       </configuration>
```

و همین طور می تواند db initializer به شرح زیر به صورت سفارشی تنظیم کنید.

```
   <?xml version="1. 0" encoding="utf-8" ?>
           <configuration>
                 <appsettings>
                       <add key="DatabaseInitializerForType
schooldatalayer. schooldbcontext,=SchoolDataLayer.
SchoolDBContext, schooldatalayer"=SchoolDataLayer"
                                 = value="SchoolDataLayer.
SchoolDBInitializer, schooldatalayer"=SchoolDataLayer" />
```

```
    </appsettings>
</configuration>
```

بنابراین بدین طریق شما می توانید استراتژی DB initialization برای Application تان به کار ببرید.