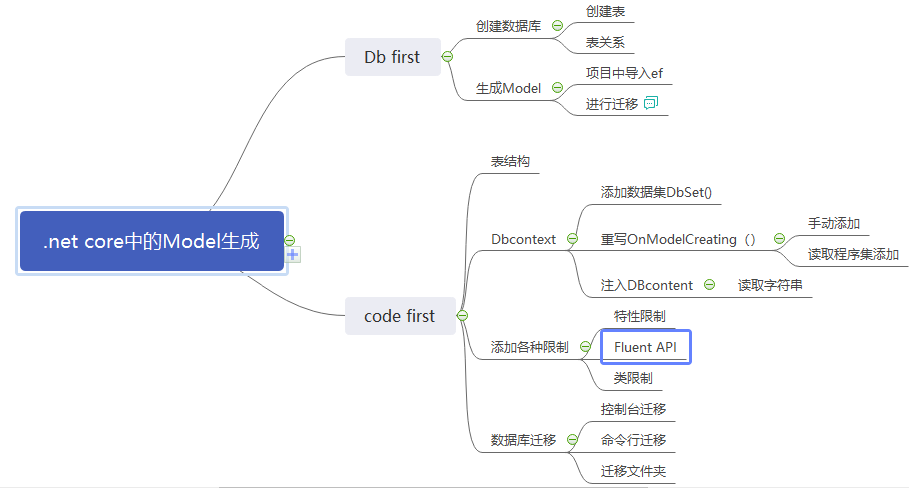
## EF Core中的DB First与Code First

### 前言：

大家都习惯在程序中生成对应的model来对数据库进行操作，所以如何快速的生成数据库表的对应model，是基础之一。总结了一下在我的认知中大概是这个结构：



### Db first方式：

先创建好对应的数据库通过数据库来生成对应的Model。

1. 创建数据库

大家可根据自己的需要生成对应的数据库，这里只做演示所以使用官方给的SqlServer数据库脚本：

CREATE DATABASE [Blogging];

GO

USE [Blogging];

GO

CREATE TABLE [Blog] (

[BlogId] int NOT NULL IDENTITY,

[Url] nvarchar(max) NOT NULL,

CONSTRAINT [PK\_Blog] PRIMARY KEY ([BlogId])

);

GO

CREATE TABLE [Post] (

[PostId] int NOT NULL IDENTITY,

[BlogId] int NOT NULL,

[Content] nvarchar(max),

[Title] nvarchar(max),

CONSTRAINT [PK\_Post] PRIMARY KEY ([PostId]),

CONSTRAINT [FK\_Post\_Blog\_BlogId] FOREIGN KEY ([BlogId]) REFERENCES [Blog] ([BlogId]) ON DELETE CASCADE

);

GO

INSERT INTO [Blog] (Url) VALUES

('http://blogs.msdn.com/dotnet'),

('http://blogs.msdn.com/webdev'),

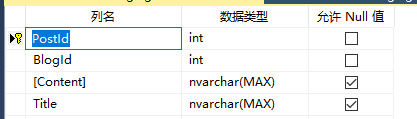
('http://blogs.msdn.com/visualstudio')

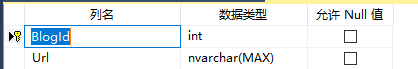
GO

执行完了之后看一下效果：



表结构（极其简单）：



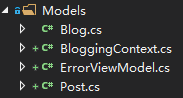


需要的数据库创建完毕！

1. 生成对应Model

在.net core核心包中已经包含了ef关于sqlserver的包，所以不需要导入任何包即可，如果你使用别的数据库则安装对应的包即可。在控制台中执行：

其中-OutputDir "Models"只将生成的文件放在Models中。



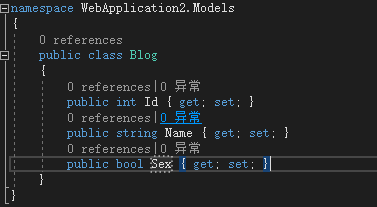
没什么问题，可以正常用了。

### Code First方式：

先写好对应的Model，然后创建到数据库。

1. 创建Model类

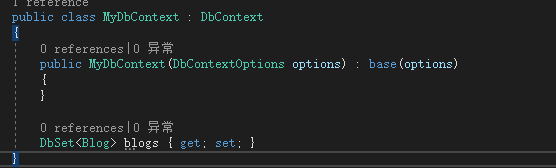
根据自己的需求写好类：



因为偷懒，所以只写了一个类。。。

1. 创建DbContext

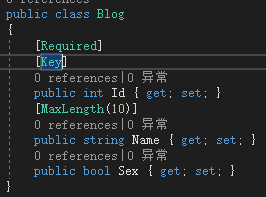
个人把DbContext理解为一个中控类，可以通过它来对数据库进行操作，这个可能理解有误。直接上图：



比较简单哈，把记得所有实体类都加进去就ok啦。

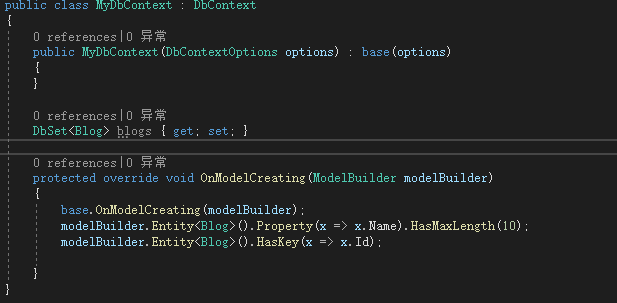
1. 特性限制类

在正常的数据库中我们会对字段加许多的限制，比如长度最长为x，不可为空，设置主键等等。如果我们使用特性来进行约束那应该是这样的：



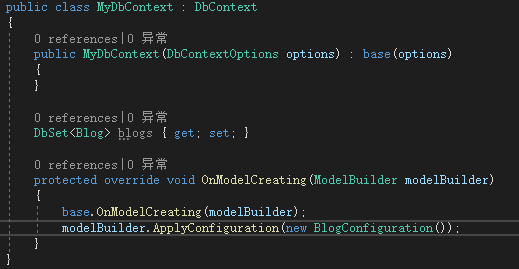
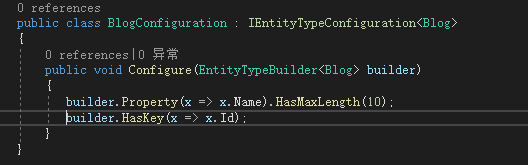
这样的话看上去比较混乱，用大佬们的话来说可能是职责不单一?应该是。总之这样是不好的。

所以我们要在DbContext中对字段进行限制，ef core有一套叫Fluent API的东西，比较方便,上图：



在OnModelCreating中加上限制之后和使用特性限制的效果是一致的。

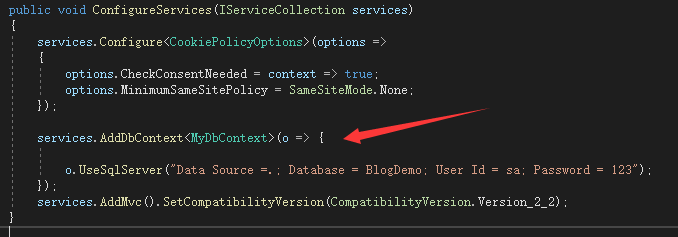
但是当你的一个表字段比较多并且限制和表关系也非常多的时候，OnModelCreating中就会有很多行，不方便阅读，所以有了限制类（我自己起的..）这个东西：



继承于IEntityConfiguration，实现Configure方法，在这个方法中使用Fluent API进行限制就可以了，这样每个表的限制对应一个类比较方便阅读。

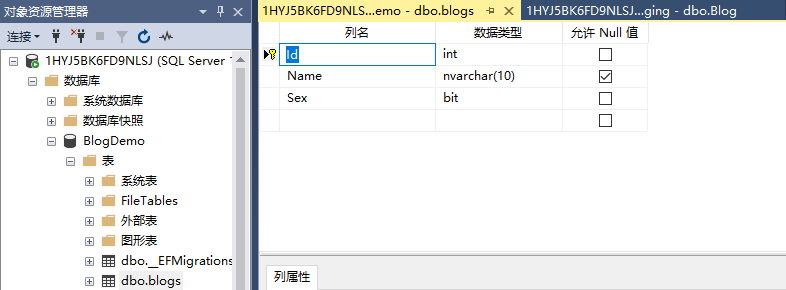
1. 迁移命令

一种是使用CLI，一种是使用nuget，命令写法不一样而已，结果都一致。当然在这之前我们需要注入一下DbContext：



CLI方式：

在对应文件夹打开命令行执行dotnet ef migrations add name和dotnet ef database update 看结果：



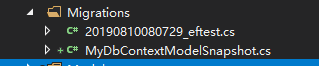
我们需要的库已经建立完成了。

Nuget方法：

在nuget中执行：add-migration name和update-database即可，结果都一样不用看了。

1. 迁移生成的文件介绍

第一次执行dotnet ef migrations add name时cli会自动创建migrations文件夹里面包含两个文件。



MyDbContextModelSnapshot文件是快照文件，记录第一次迁移时的数据库结构，以日期命名的文件为修改文件，记录每一次对数据库修改了哪些地方，如果迁移之后更新数据库之前发现还需要更改可以使用remove进行回滚。

具体回滚操作可使用：dotnet ef migrations remove

### 结语

只有在平时使用的时候，因为各种原因会遇到各种各样的奇葩问题，这里就不列举了，遇到的时候就得面向Search开发了，一般都是粗心错误，一般都能找到解决方案。

今天就先写到这里，希望可以帮到大家，如果帮不到就算我记笔记了。。。

### 相关资料

FluentAPI文档：<https://www.learnentityframeworkcore.com/configuration/fluent-api>

EFCore创建模型部分：<https://docs.microsoft.com/zh-cn/ef/core/modeling/>

官方案例：<https://docs.microsoft.com/en-us/ef/core/get-started/aspnetcore/new-db?tabs=visual-studio>