**.NETCore开源快速开发框架发布(NETCore+AdminLTE版)**

# 引言

上星期发布的.NET452+AdminLTE版框架得到了大家的大力支持，目前GitHub已获近100星，鄙人十分高兴能够帮助大家。其中有不少关心.NET Core的支持情况，因为未来的.NET平台是属于.NET Core的，为此，鄙人花了一星期将代码移植到.NET Core2.1，不负众托，在此推出NETCore+AdminLTE版。

# 简介

本框架旨在为.NET开发人员提供一个Web后台快速开发框架，采用本框架，能够极大的提高项目开发效率。

本版本框架采后端采用基于.NET Core的ASP.NET Core2.1，前端从原来的Easyui升级为基于Bootstrap的AdminLTE，支持响应式布局，提高用户体验（提高逼格）。

框架主要功能如下表：

|  |  |  |
| --- | --- | --- |
| **功能** | **详细描述** | **支持情况** |
| 用户登录、密码修改 | 用户登录、密码修改 | ✔ |
| 系统用户管理 | 系统用户管理 | ✔ |
| 角色管理 | 角色管理 | ✔ |
| 权限管理 | 基于用户的权限、基于角色的权限、菜单权限、操作权限（按钮权限）、接口权限、数据权限 | ✔ |
| 系统日志 | 支持使用ElasticSearch记录日志 | ✔ |
| **快速开发** | **通过数据库直接生成实体层、业务逻辑层、视图层代码， 无需编码即可实现CURD** | ✔ |
| 数据库操作封装 | 使用基于EF的仓储模式、封装常用的CURD，支持单库事物和分布式事物 | ✔ |
| 多数据库支持 | 使用基于EF的仓储模式，支持各大主流关系型数据库（SQLServer、MySQL、Oracle、PostgreSQL） | ✔ |
| 缓存支持 | 支持系统自带缓存和Redis缓存、封装操作接口、简单易用 | ✔ |
| 分布式Session支持 | 禁用自带Session，使用自定义Session，一键开启分布式Session | ✔ |
| 页面响应式 | 基于Bootstrap，支持响应式布局，移动端能够轻松访问 | ✔ |
| 其它各种帮助类库及插件 | 其它各种帮助类库及插件 | ✔ |

后台展示如下：



版本参照请看下表：

|  |  |  |
| --- | --- | --- |
| **.NET版本** | **前端UI** | **地址** |
| .NET Core2.1 | AdminLTE | <https://github.com/Coldairarrow/Colder.Fx.Core.AdminLTE> |
| .NET4.52 | AdminLTE | <https://github.com/Coldairarrow/Colder.Fx.Net.AdminLTE> |
| .NET4.52 | Easyui | <https://github.com/Coldairarrow/Coldairarrow.Fx.Net.Easyui.GitHub> |
| .NET Core2.1 | Easyui | <https://github.com/Coldairarrow/Coldairarrow.Fx.Core.Easyui.GitHub> |
| .NET4.0 | Easyui | <https://github.com/Coldairarrow/Coldairarrow.Fx.Net40.Easyui.GitHub> |

# 基础准备

## 开发环境要求

操作系统：Windows 10

开发工具：Visual Studio 2017

SDK：安装.NET Core2.1及以上

数据库：SQLServer2008 R2及以上

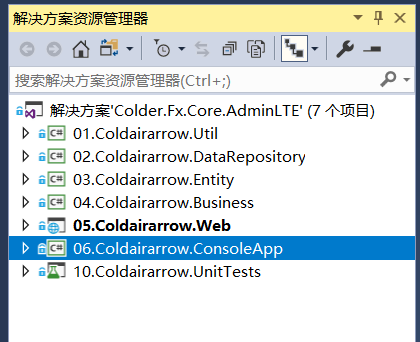
## 基础数据库构建

使用本框架需要构建基础数据库，具体步骤如下：

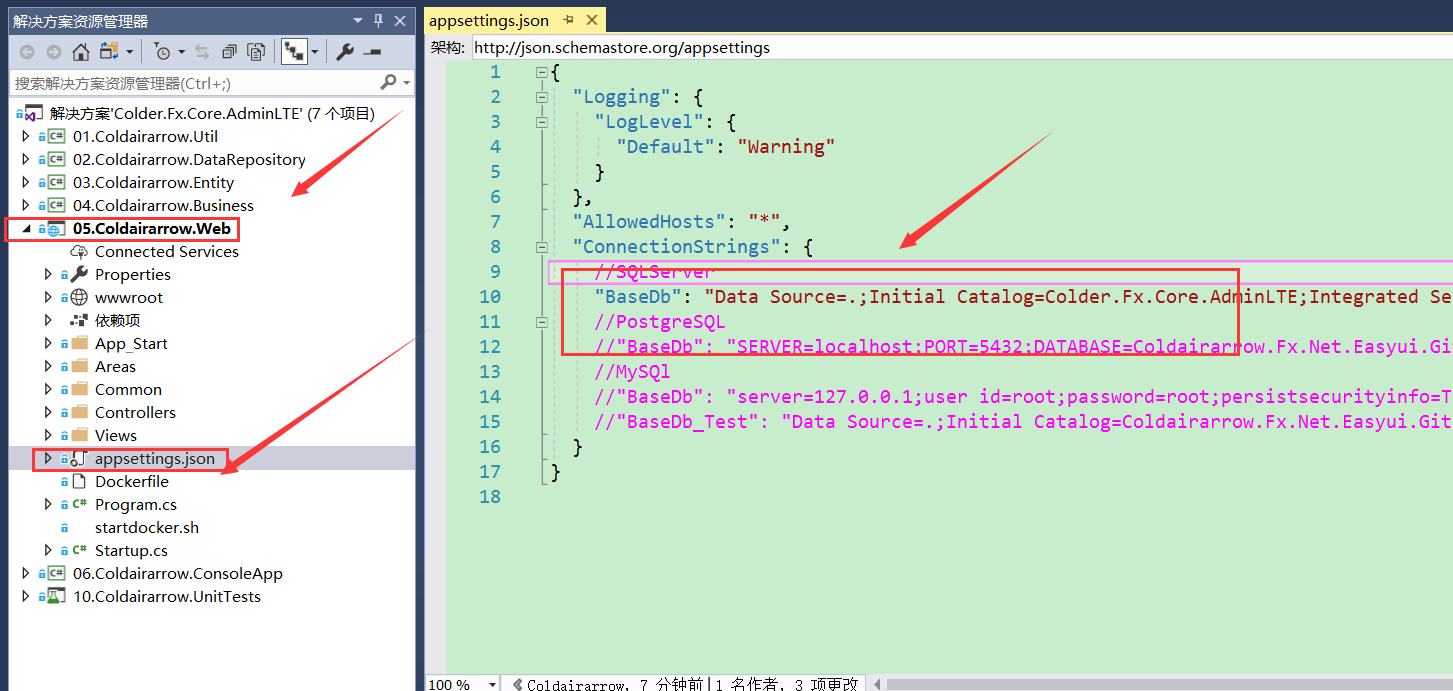
创建基础数据库的Sql脚本文件在：/docs/初始化文件/db.sql，在数据库中运行db.sql脚本即可创建数据库: Colder.Fx.Core.AdminLTE

若Sql运行出错，请直接使用同目录下的db.bak还原数据库

打开src目录下Colder.Fx.Core.AdminLTE的解决方案，如下图



如下图所示依次展开05.Coldairarrow.Web=>appsettings.json，配置数据库连接字符串，name不用修改，connectionString改为上述创建的数据库（若不清楚数据库连接字符串请自行百度搜索教程）

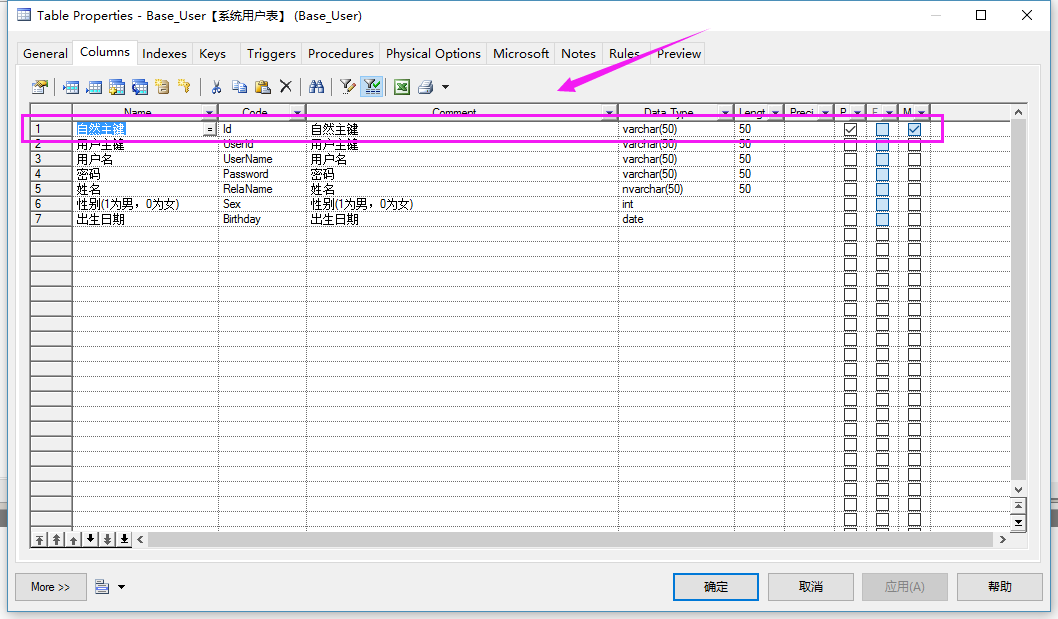


自此基础数据库配置完成。

## 数据库设计规范

由于本框架支持自动生成代码的核心功能，此功能是根据数据库的表结构来生成代码的

因此规定每张表都有一个主键，列名为Id，类型为字符串，实际添加数据时默认使用GUID的字符串形式，表中的每个列都需要有描述信息（建议这样操作，若不按照这个标准则需要一些额外的改动才能够成功运行）。



## 运行

请先还原Nuget包

然后将05.Coldairarrow.Web设为启动项目，成功运行即可进入以下页面



# 详细教程

## 代码架构

项目代码分层，详情见源码中的说明书，博客园放会被判为重复文章，敬请谅解。

## 功能架构

### 全局配置

在01.Coldairarrow.Util中的GlobalSwitch类中，设置了各个参数，其中RunModel需要重点关注一下，若RunModel==RunModel.LocalTest，则系统会直接跳过登录，默认使用Admin超级管理员登录，其它参数请看注释。

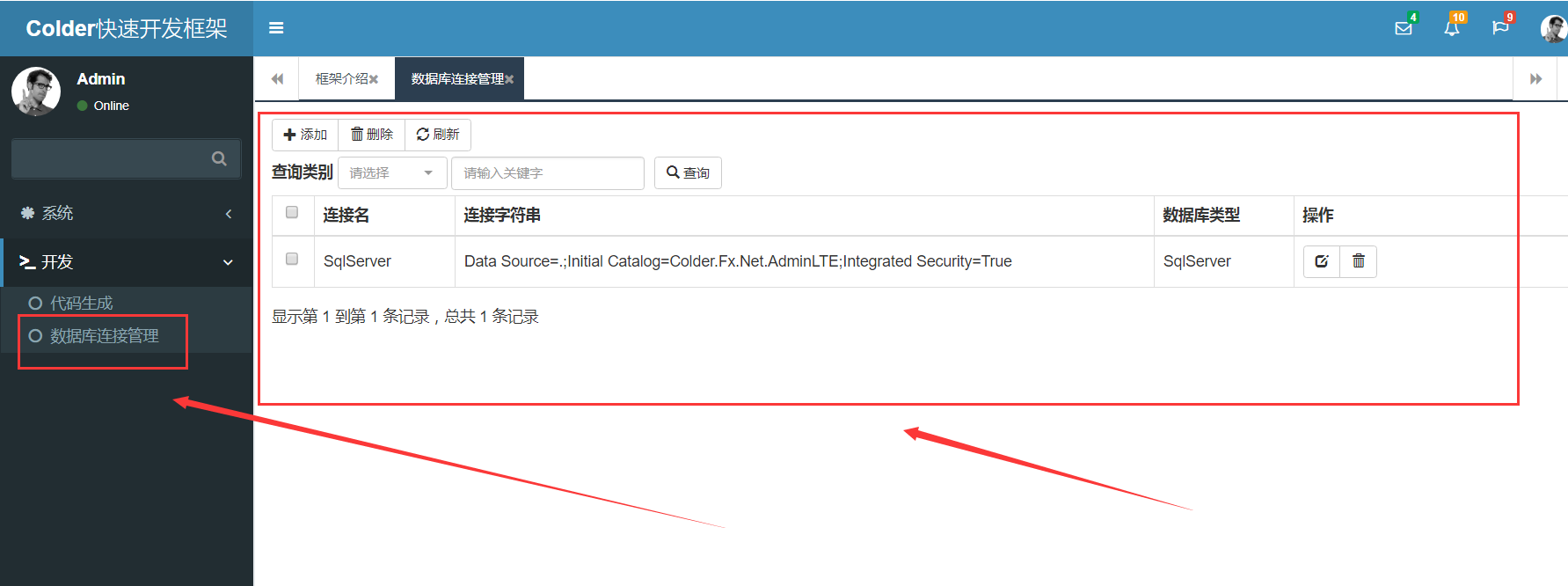
### 快速开发

使用此功能请确保GlobalSwitch.RunModel= RunModel.LocalTest

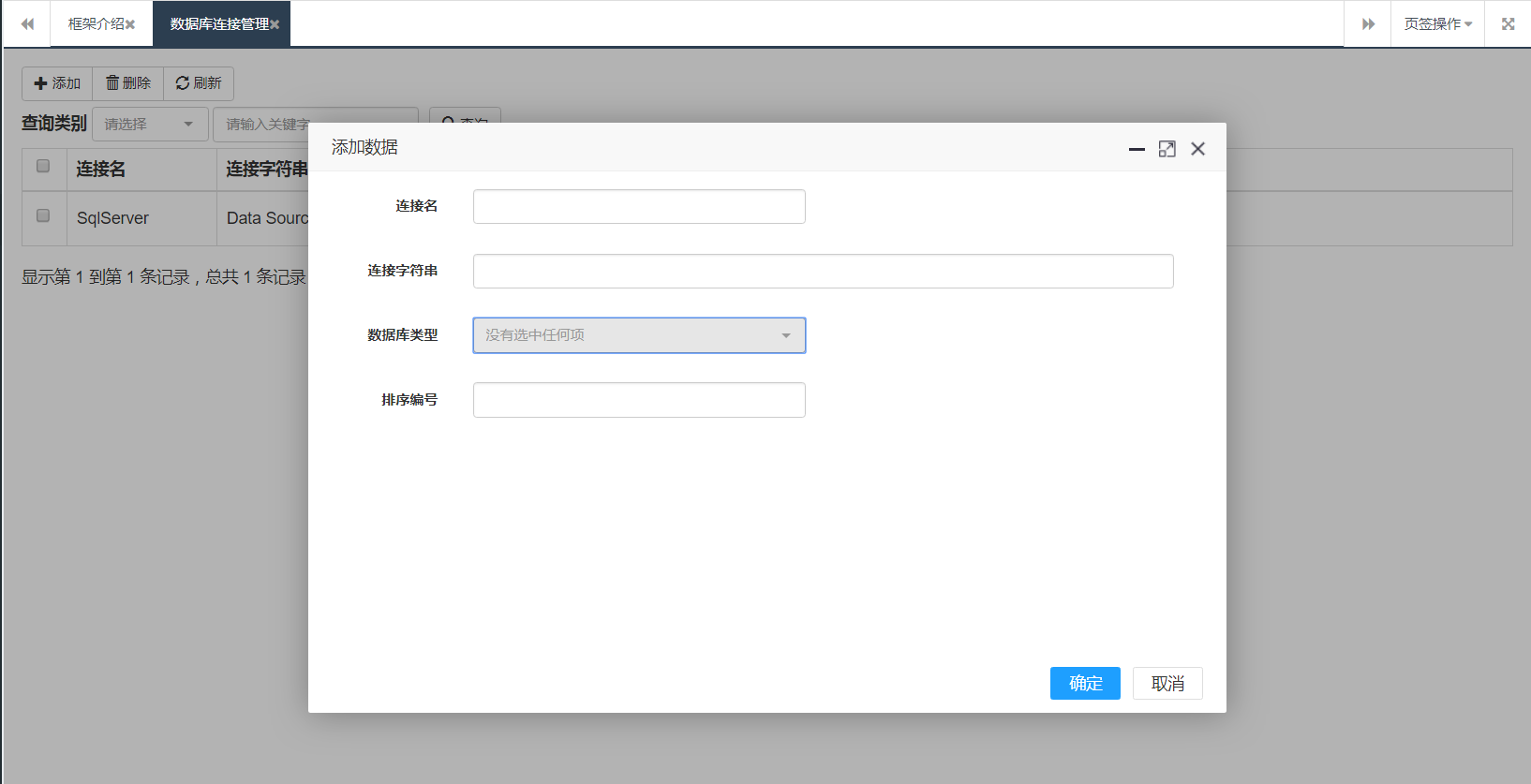
此功能为本框架的核心功能，能够自动生产完整的可运行代码，具体使用如下：

首选需要有数据库源，因为代码生成是根据数据库表来生成的。

菜单：开发=>快速开发=>数据库连接管理



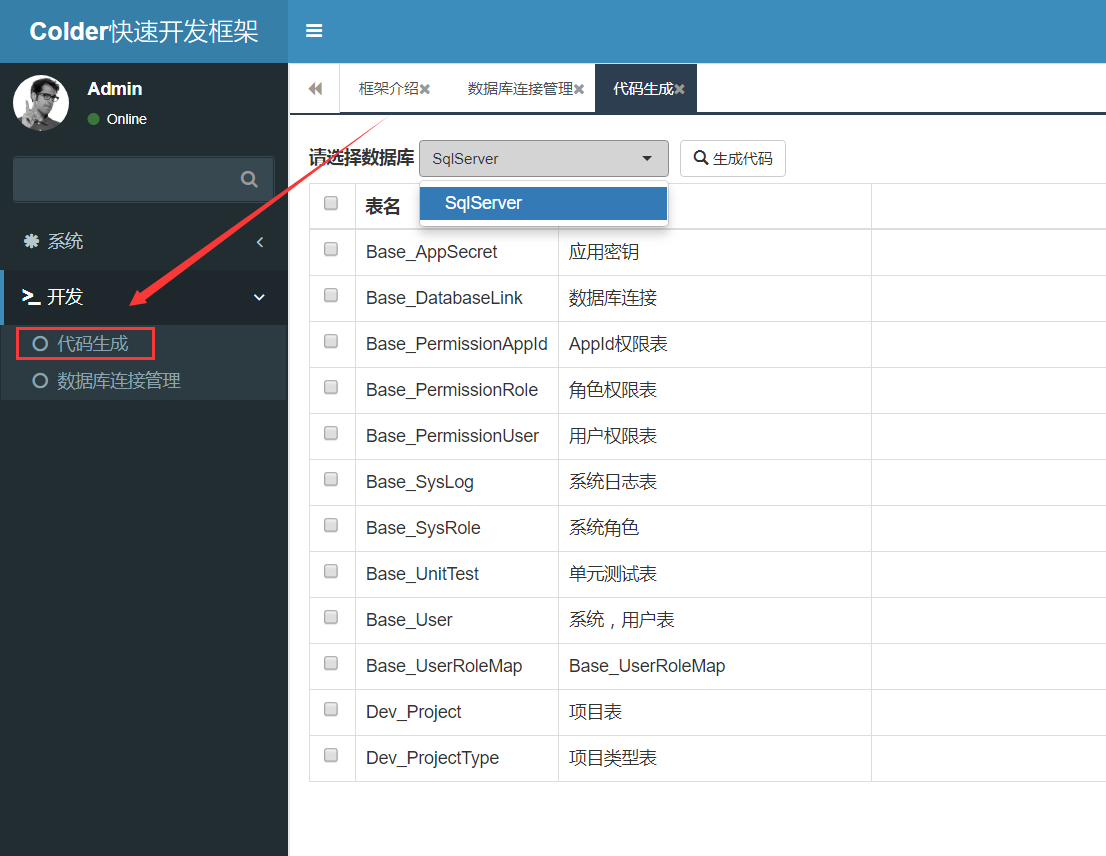
若列表中没有目标数据源，则添加数据库连接



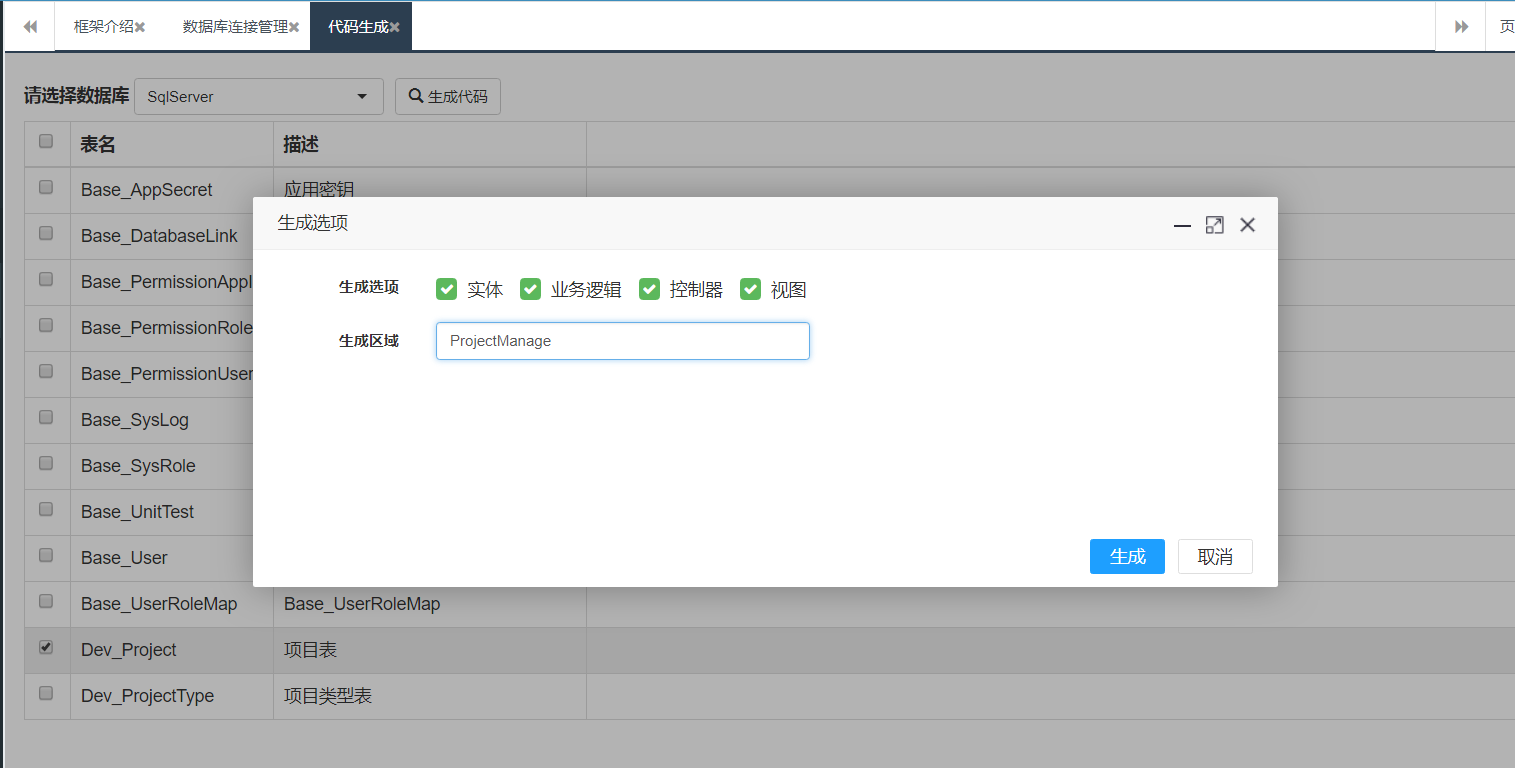
数据连接名、连接字符串、数据库类型即可。添加完成后即可看到连接字符串信息。

有了数据库连接之后，即可进行代码生成。

菜单：开发=>快速开发=>代码生成

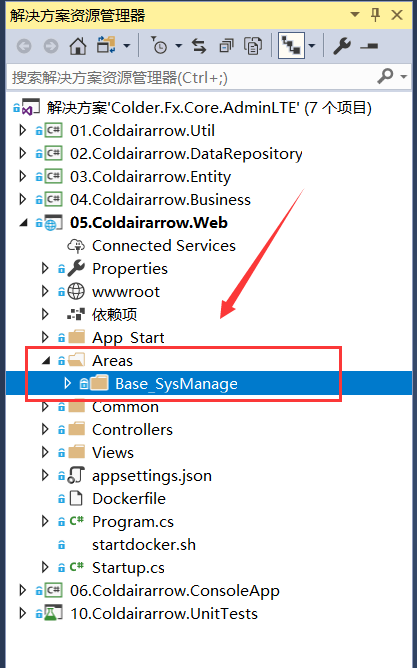


选择数据库，然后勾选需要生成代码的数据库表，点击生成代码会弹出生成选项（这里暂时只能勾选Dev\_Project，其余表全是系统基础表，不要勾选，否则会被覆盖，导致异常，请勾选自己的业务表进行生成）：

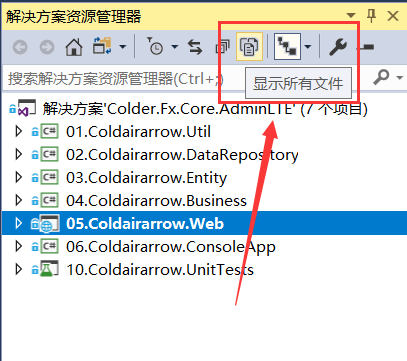


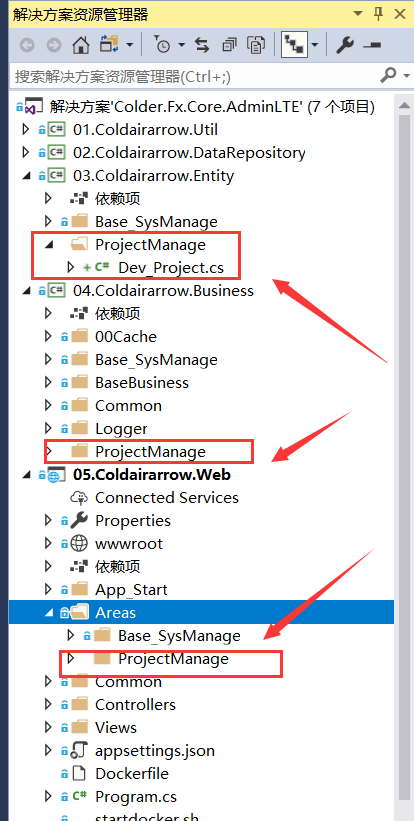
生成选项中可以选择需要生成的类型，可以默认生成实体层、业务逻辑层、控制器和视图。

生成区域对应MVC中的Areas，请按需填入（必填）



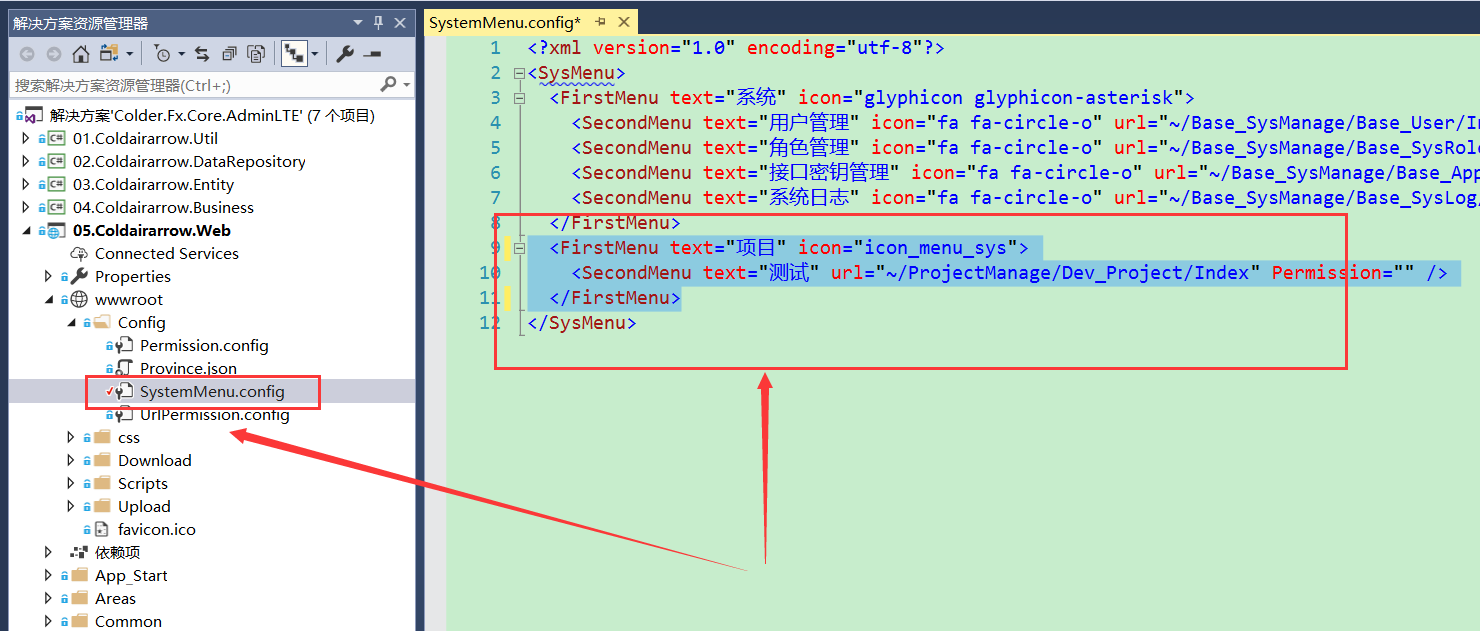
这里示例填写ProjectManage，点击生成按钮，即可完成代码生成。生成后的代码在项目解决方案中，将代码文件包括进入项目（.NET Core项目默认会自动包括新生成的文件,若没包括请手动包括进项目中）



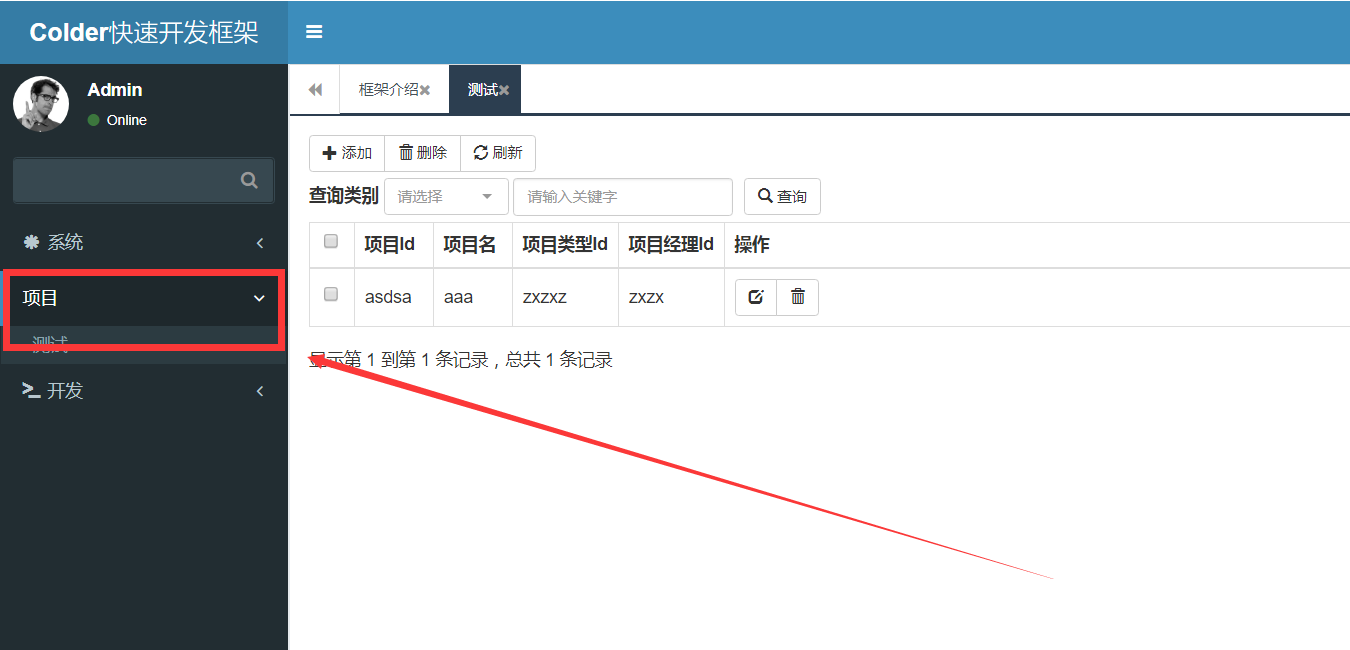


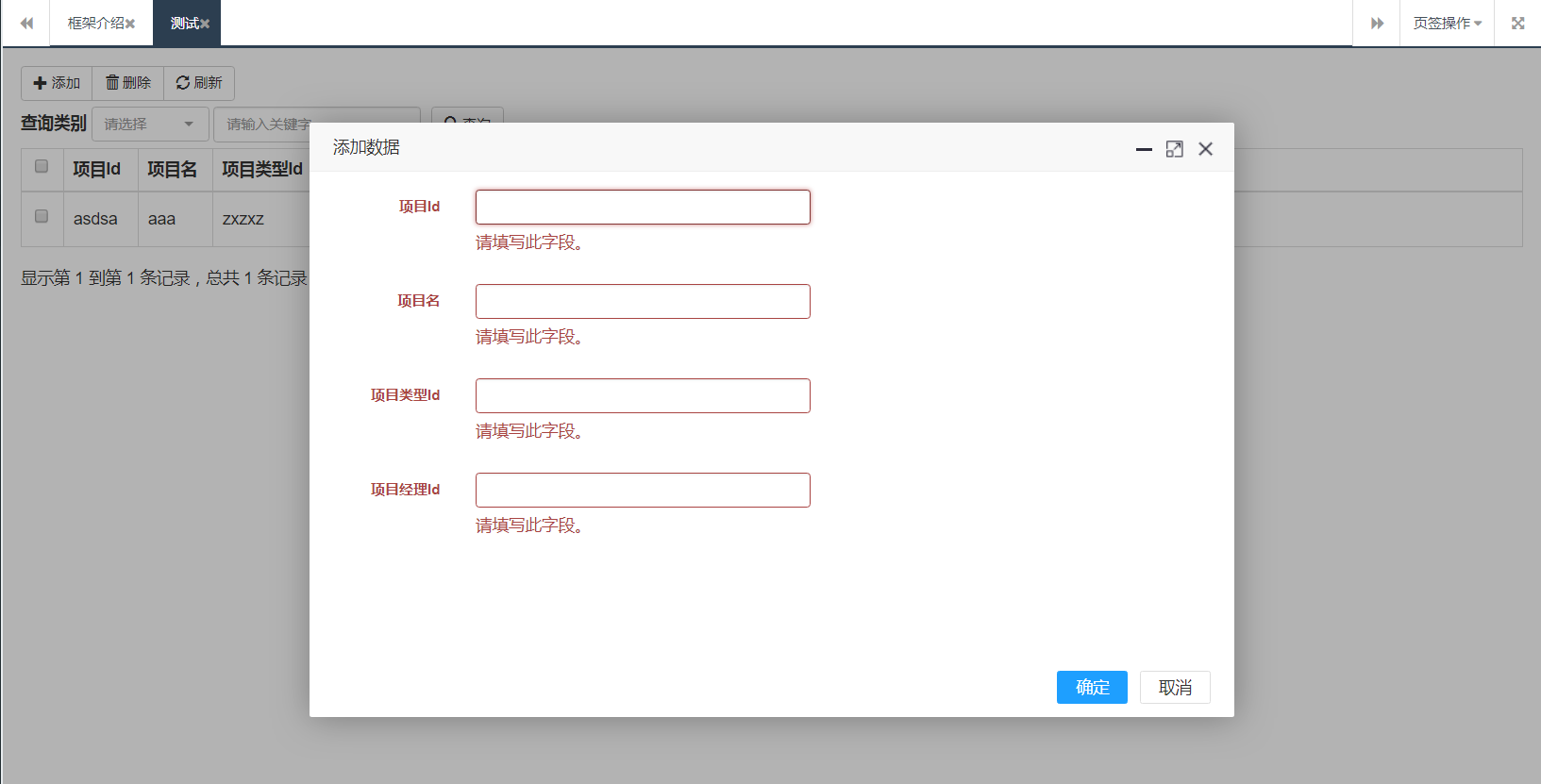
右键新的文件夹，包括在项目中

由于是新生成的代码，所以才配置新的菜单



如上图，在Web项目中的wwwroot/Config/SystemMenu.config中配置菜单，模仿原有菜单即可，其中url是指页面的路径，Permission是指若需要访问此菜单需要的权限（对应权限配置），若没有此权限，则菜单也中不会显示此菜单，修改完成后重新编译生成（权限相关模块进行了缓存，重新生成会清缓存），重新运行即可看到新的项目菜单如下：

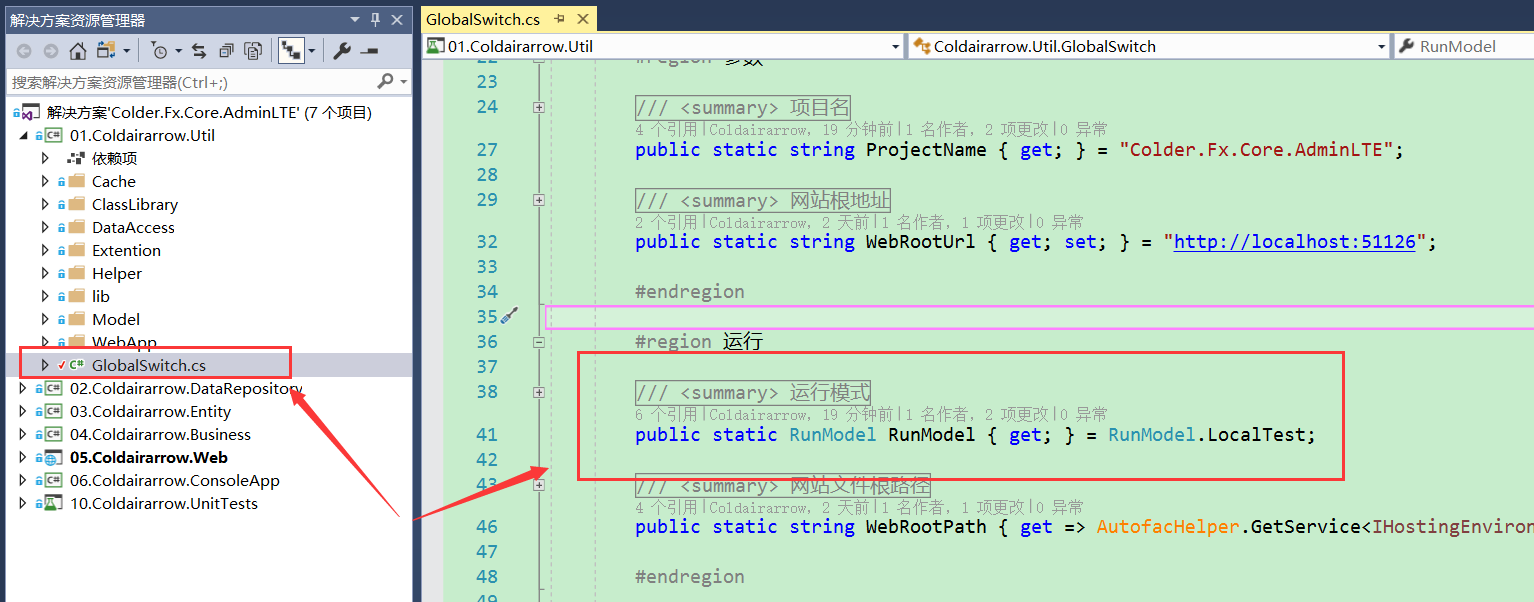




整个代码生成过程，无需编写代码即可完成一张表的CRUD，当然需要根据具体业务中进行相应的修改，本次示例中字段比较少，但是当一张表的字段很多时，那么此功能能够将开发效率提高几个档次。

### 管理员登录

若要使用登录功能，请将GlobalSwitch中的RunModel改为RunModel.Publish





默认超级管理员账号为：Admin

密码为：123456

### 系统用户管理

管理系统登录的用户

菜单：系统=>用户管理，如下页面



点击右侧设置权限，可以设置用户权限，详情见<权限管理>模块

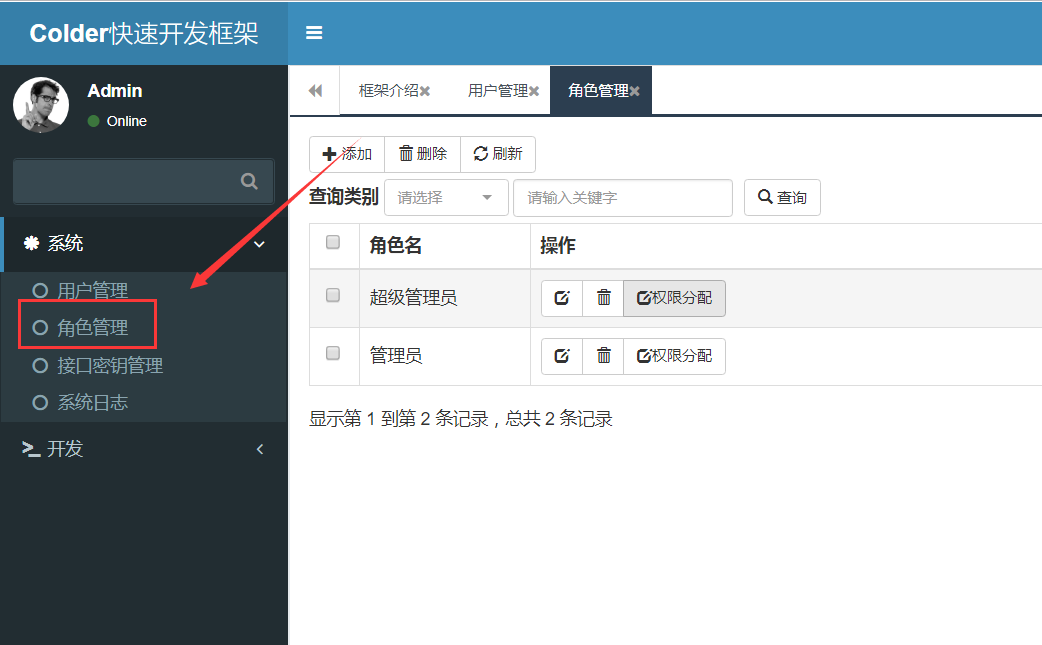


具体权限相关配置见权限管理模块

### 系统角色管理

管理系统角色，角色是权限的载体，合理分配角色有利于权限管理

菜单：系统=>角色管理



操作中可以设置角色的权限，详情见<权限管理>模块

### 权限管理

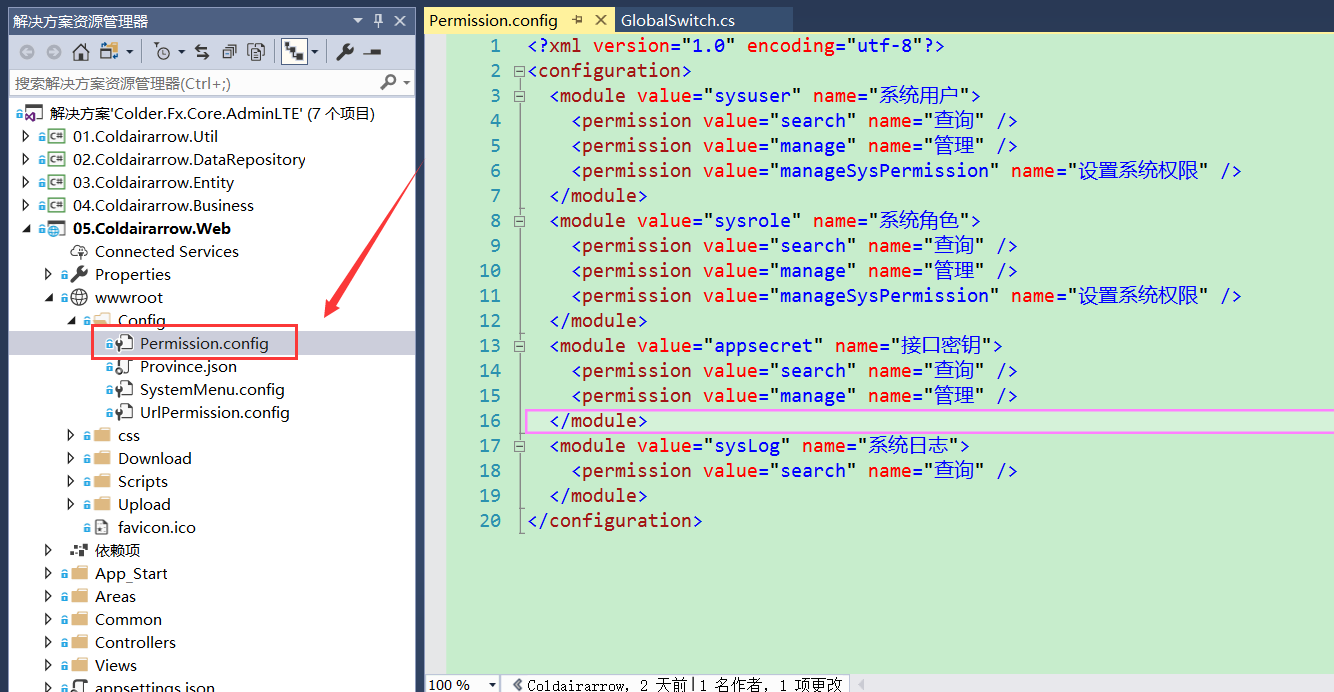
一般情况下，后台管理系统多少会涉及权限管理，因此本框架提供了一个灵活、高效、简洁的权限管理系统。

首先，权限分为两种，即操作权限和数据权限，其中操作权限报货系统用户权限和AppId权限，系统用户权限就是指操作用户具备哪些权限，而当对外提供Api接口时，为了保证接口的安全性（若不在意可忽略），通常会提供接口签名算法，其中AppId和AppSecret是必备的，通过对AppId设置权限，即可控制接口的权限。数据权限比较复杂，若采用纯SQL方式，那么会更加复杂，本框架全程采用EF作为ORM框架，通过对IQueryable<T>进行过滤，即可完成数据权限控制。

用户权限：若对每个用户都设置对应的权限，那么工作量无疑是十分巨大的，因此引入了角色的概念，角色是权限的集合载体，那么属于此角色的用户就继承了角色的权限，当然某些特殊用户需要拥有自己的不属于角色的特殊权限，因此最终用户拥有的权限就是自己的权限和所属角色权限的并集。

权限使用：

权限定义：

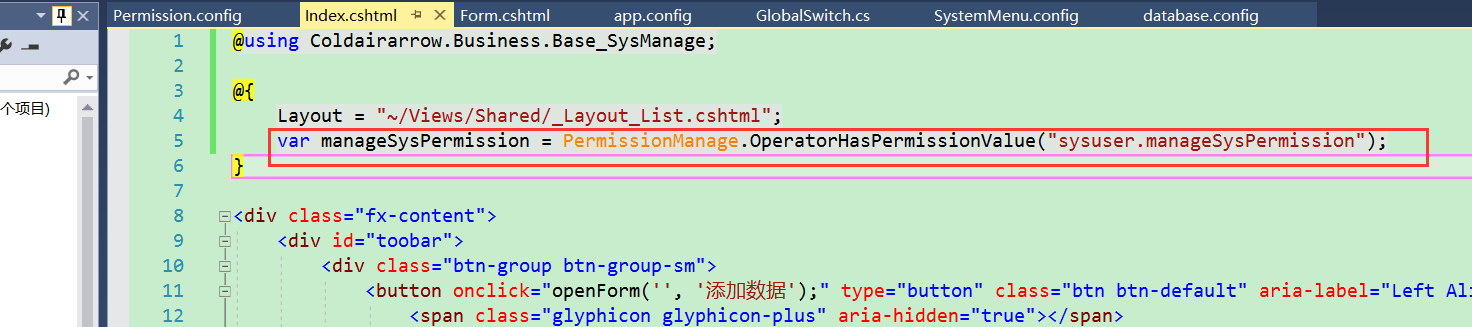


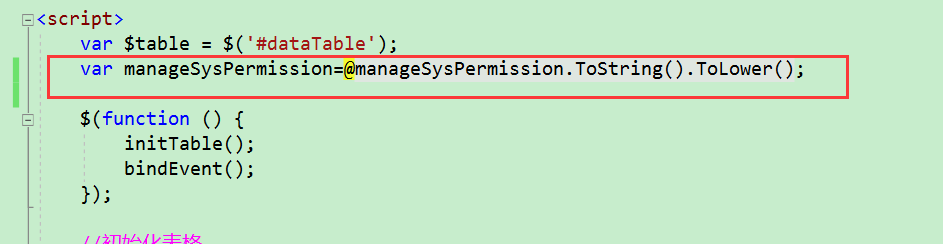
如上图，在Permission.config中定义了各个权限

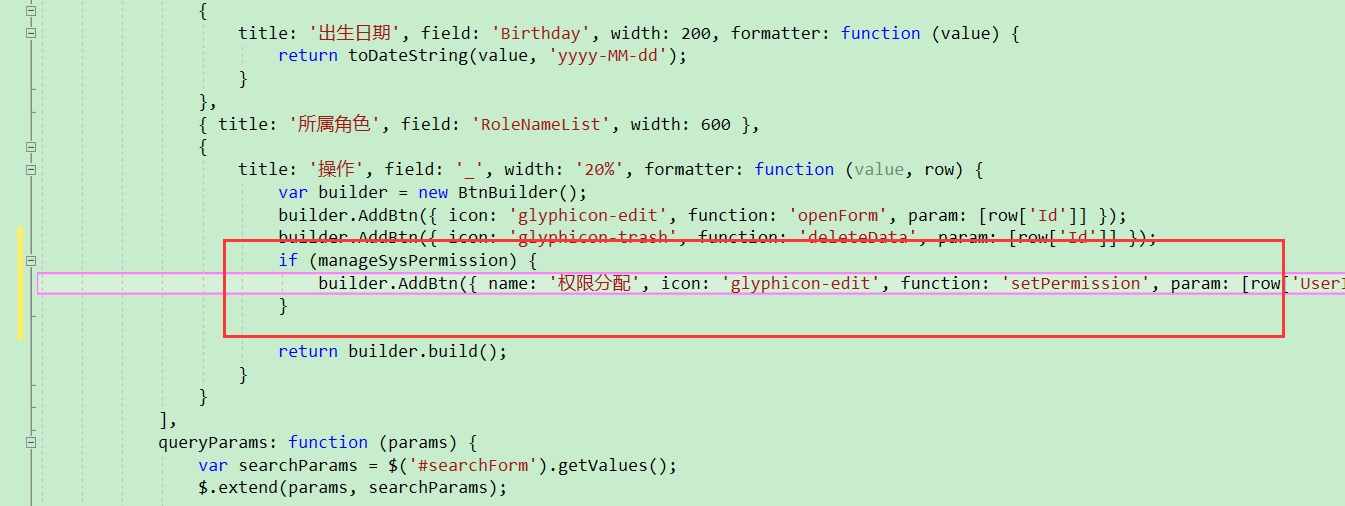
权限配置：

在系统用户管理和系统角色管理中可以设置用户和角色的权限，把需要的权限勾选即可。

权限使用：







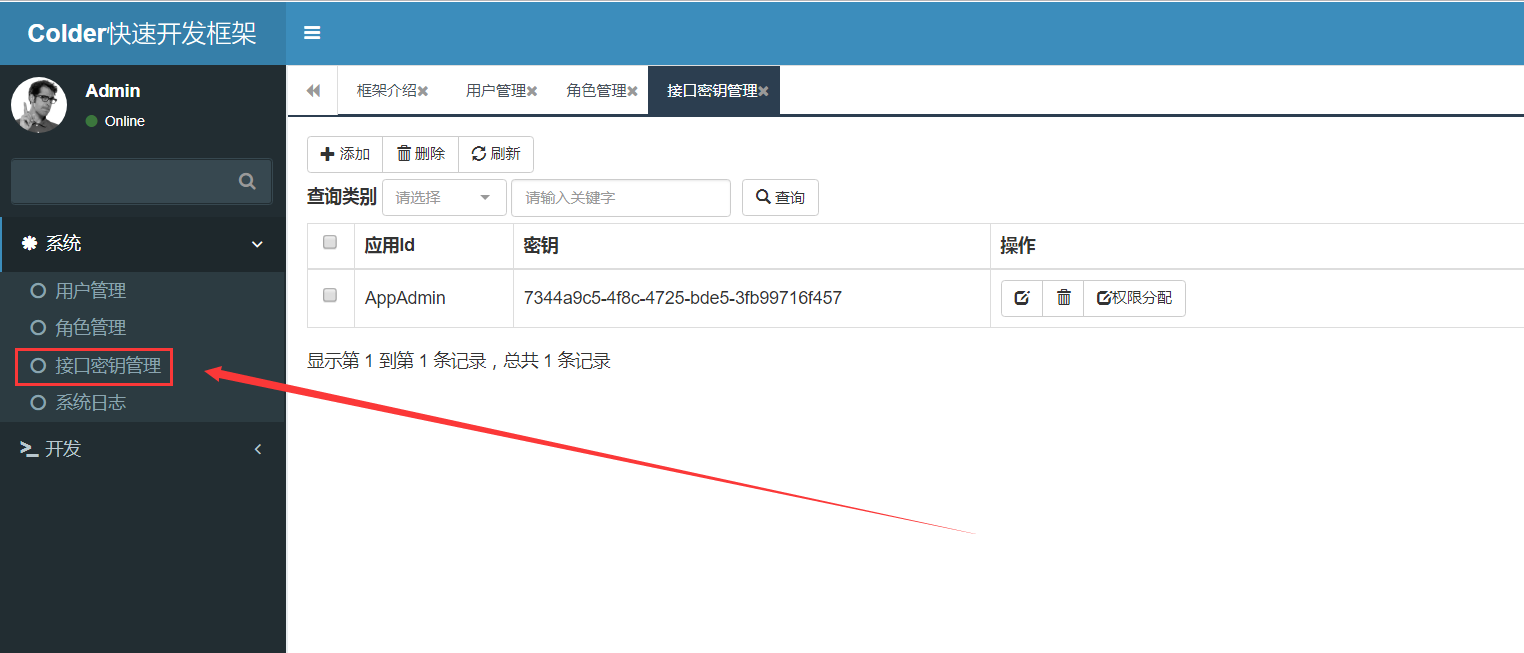
如上图所示，在需要控制权限的页面中，调用方法：PermissionManage.OperatorHasPermissionValue("sysuser.manage")

这个方法是判断操作者用户是否含有sysuser.manage权限值，其中sysuser是指Permission.config中定义的module的value属性，manage是指permission中的value属性，用.连接即是最终权限值。

更详细的使用方式，请参考源代码。

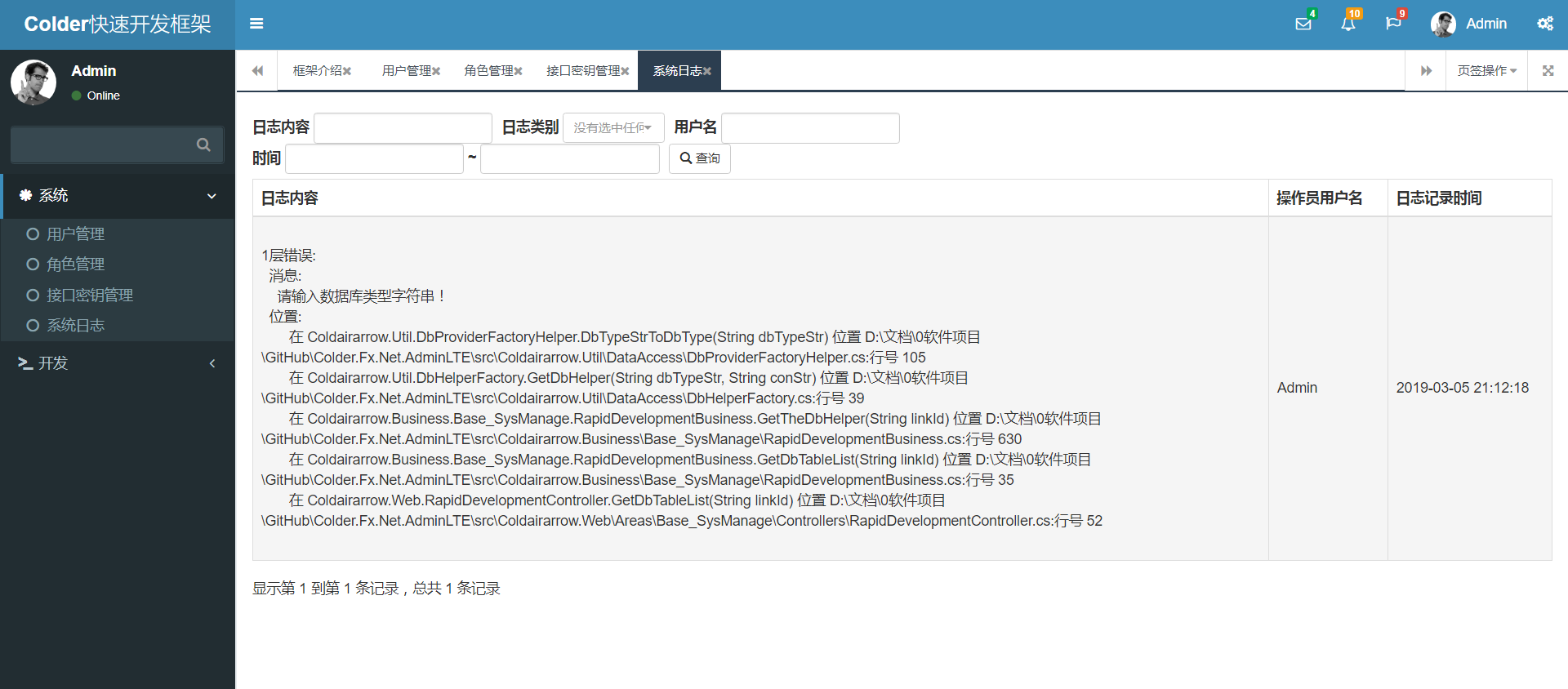
### 接口秘钥管理

菜单：系统=>接口秘钥管理



### 系统日志

菜单：系统=>系统日志

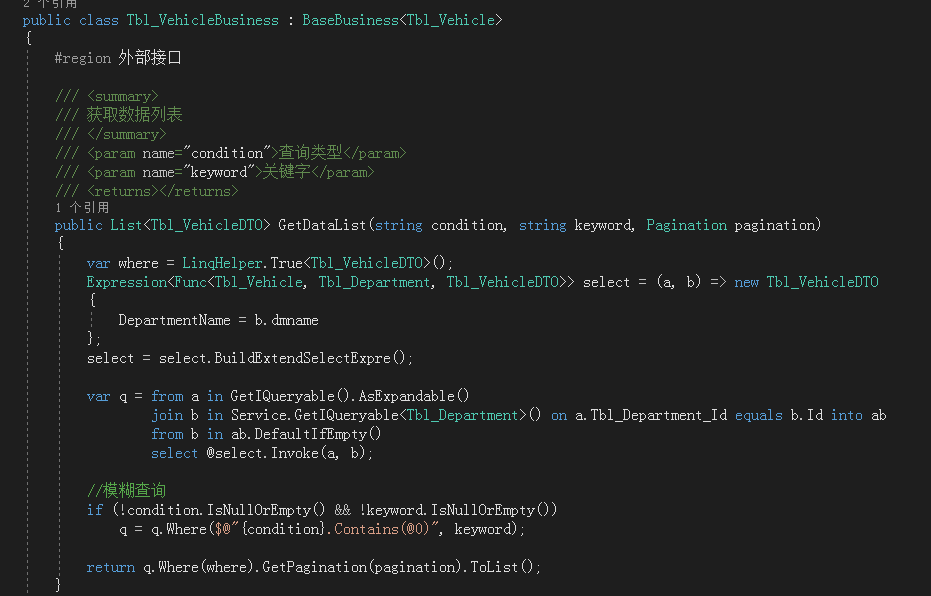


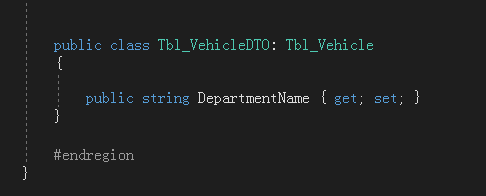
# 常见疑问

## 如何进行联表查询？

框架使用EF+LINQ进行联表操作，核心在于对IQueryable<T>的使用，另可网上搜EF+LINQ的相关教程。

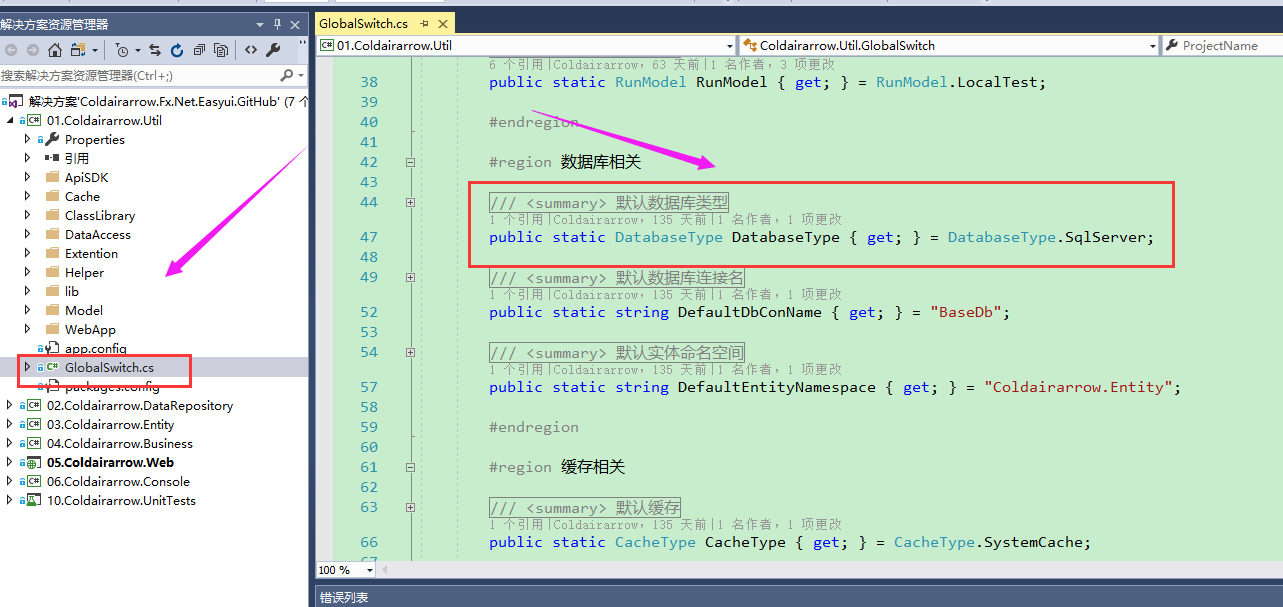
示例如下图：





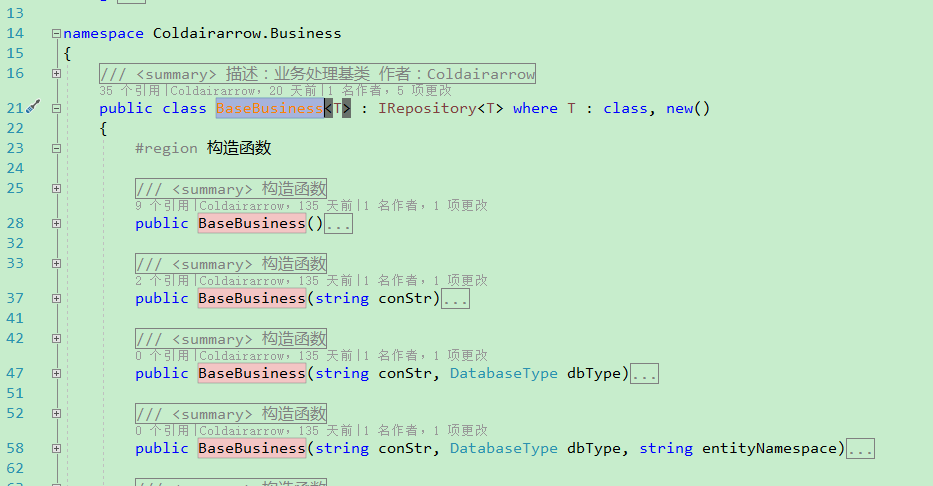
## 如何切换数据库？

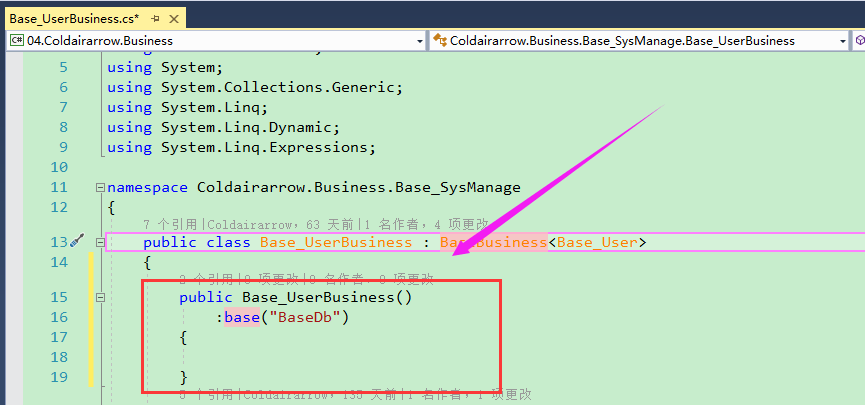
在01.Coldairarrow.Util项目中的GlobalSwitch，将DatabaseType改为需要的即可，对应的数据库连接字符串当然也要改为对应数据库的



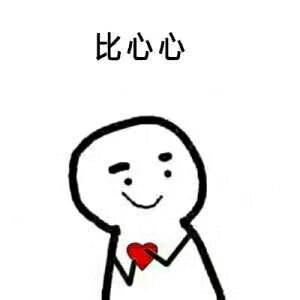
## 如何使用别的数据库（即非默认的系统数据库）

在具体的Business类中重写父类BaseBusiness的构造函数即可，按照自己的需求重写对应的构造函数,同时需要确保数据库连接字符串已添加





# 结语

欢迎使用本框架，若觉得不错，请比心

Github:<https://github.com/Coldairarrow>，请Statrt

博客园：<https://www.cnblogs.com/coldairarrow/>

QQ群：373144077

本人将会对这个快速开发框架不断完善与维护，希望能够帮助到各位

若遇到任何问题或需要技术支持，请联系我。

---------------------学习永无止境，技术永无上限，代码就是艺术-----------------------