

第22期-自定义迁移操作

2019年5月17日 23:13

自定义迁移操作

MigrationBuilder API 可在迁移过程中执行许多不同的操作，但它并不详尽，可使用 Sql() 方法更加灵活。

MigrationBuilder Class

```
static MigrationBuilder CreateUser(  
    this MigrationBuilder migrationBuilder,  
    string name,  
    string password)  
=> migrationBuilder.Sql($"CREATE USER {name} WITH PASSWORD ' {password}' ;");
```

若要支持多个提供程序，下面是支持 Microsoft SQL Server 和 PostgreSQL 示例。

```
static MigrationBuilder CreateUser(  
    this MigrationBuilder migrationBuilder,  
    string name,  
    string password)  
{  
    switch (migrationBuilder.ActiveProvider)  
    {  
        case "Npgsql.EntityFrameworkCore.PostgreSQL":  
            return migrationBuilder  
                .Sql($"CREATE USER {name} WITH PASSWORD ' {password}' ;");  
  
        case "Microsoft.EntityFrameworkCore.SqlServer":  
            return migrationBuilder  
                .Sql($"CREATE USER {name} WITH PASSWORD = ' {password}' ;");  
    }  
  
    return migrationBuilder;  
}
```

MigrationOperation

使用一个单独的项目

支持在其它程序集中存储迁移快照，借助此方式，可维护多个版本的迁移文件，分别用于开发和测试。

1. 创建一个新的类库。
2. 添加 DbContext 所在程序集的引用。

3. 将迁移和模型快照文件移动到这个类库。

4. 配置迁移程序集。

```
options.UseSqlServer(connectionString, x => x.MigrationsAssembly("MyApp.Migrations"));
```

5. 更新的类库的输出路径

```
<PropertyGroup>
```

```
  <OutputPath>..\MyStartupProject\bin\$(Configuration)\</OutputPath>
```

```
</PropertyGroup>
```

```
Add-Migration NewMigration -Project MyApp.Migrations
```

```
dotnet ef migrations add NewMigration --project MyApp.Migrations
```