

# 第11期-基于EfCore存储认证授权数据

2020年6月16日 14:34

## 数据存储

两种类型的数据：配置数据（客户端，资源，范围） + 运行数据（授权码，刷新令牌）

## 模板创建新项目集成

```
dotnet new -i IdentityServer4.Templates
```

```
dotnet new is4empty
```

```
dotnet new is4ui
```

```
dotnet new is4ef
```

## 现有项目集成

```
Install-package IdentityServer4.EntityFramework
```

```
Install-package Microsoft.EntityFrameworkCore.SqlServer
```

## 数据库架构更改和使用EF迁移

您可以在IdentityServer4.EntityFramework.Storage存储库中找到SqlServer 的 [最新SQL脚本](#)。

也可以使用 EfCore 迁移命令进行。

## 配置数据存储机制

```
const string connectionString = @"Data
Source=(LocalDb)\MSSQLLocalDB;database=IdentityServerDB";

var migrationsAssembly = typeof(Startup).GetTypeInfo().Assembly.GetName().Name;

services.AddIdentityServer()
    .AddTestUsers(TestUsers.Users)
    .AddConfigurationStore(options =>
    {
        options.ConfigureDbContext = b => b.UseSqlServer(connectionString, sql =>
            sql.MigrationsAssembly(migrationsAssembly));
    })
    .AddOperationalStore(options =>
```

```
{
    options.ConfigureDbContext = b => b.UseSqlServer(connectionString, sql =>
sql.MigrationsAssembly(migrationsAssembly));
})
.AddDeveloperSigningCredential();
```

## 添加迁移支持

Install-package Microsoft.EntityFrameworkCore.Tools

Add-Migration InitialIdentityServerPersistedGrantDbMigration -c PersistedGrantDbContext -o Data/Migrations/IdentityServer/PersistedGrantDb

Add-Migration InitialIdentityServerConfigurationDbMigration -c ConfigurationDbContext -o Data/Migrations/IdentityServer/ConfigurationDb

## 初始化数据库并添加演示数据

```
private void InitializeDatabase(IApplicationBuilder app)
{
    using (var serviceScope =
app.ApplicationServices.GetService<IServiceScopeFactory>().CreateScope())
    {
        serviceScope.ServiceProvider.GetRequiredService<PersistedGrantDbCon
text>().Database.Migrate();

        var context =
serviceScope.ServiceProvider.GetRequiredService<ConfigurationDbContext>();
        context.Database.Migrate();
        if (!context.Clients.Any())
        {
            foreach (var client in Config.Clients)
            {
                context.Clients.Add(client.ToEntity());
            }
            context.SaveChanges();
        }

        if (!context.IdentityResources.Any())
        {
            foreach (var resource in Config.Ids)
            {
                context.IdentityResources.Add(resource.ToEntity());
            }
            context.SaveChanges();
        }

        if (!context.ApiResources.Any())
```

```

        {
            foreach (var resource in Config.Apis)
            {
                context.ApiResources.Add(resource.ToEntity());
            }
            context.SaveChanges();
        }

        if (!context.ApiScopes.Any())
        {
            foreach (var scope in Config.ApiScopes)
            {
                context.ApiScopes.Add(scope.ToEntity());
            }
            context.SaveChanges();
        }
    }
}

public void Configure(IApplicationBuilder app)
{
    // this will do the initial DB population
    InitializeDatabase(app);

    // the rest of the code that was already here
    // ...
}

```

