

# 第03期-使用用户密码保护API接口

2020年6月9日 19:34

## 自定义身份标识范围

```
public static IEnumerable<IdentityResource> GetIdentityResources()
{
    var customProfile = new IdentityResource(
        name: "custom.profile",
        displayName: "Custom profile",
        claimTypes: new[] { "name", "email", "status" });

    return new List<IdentityResource>
    {
        new IdentityResources.OpenId(),
        new IdentityResources.Profile(),
        customProfile
    };
}
```

## JWT令牌格式

JSON Web Token 是一种开放的行业标准 RFC 7519方法，用于在双方之间安全地表示声明。

## 获取令牌值

```
string token = HttpContext.GetTokenAsync("access_token").Result;
```

## JWT身份令牌中的申明转换为微软申明

```
JwtSecurityTokenHandler.DefaultInboundClaimTypeMap.Clear();
```

ClaimTypeMapping.cs

## 扩展用户密码验证逻辑

```
public class ResourceOwnerPasswordValidator : IResourceOwnerPasswordValidator
{
    public Task ValidateAsync(ResourceOwnerPasswordValidationContext context)
    {
        if (context.UserName == "username" && context.Password == "password")
        {
            context.Result = new GrantValidationResult(context.UserName,
                GrantType.ResourceOwnerPassword);
        }
    }
}
```

```

        }

        return Task.CompletedTask;
    }
}

```

```
services.AddIdentityServer().AddResourceOwnerValidator<ResourceOwnerPasswordValidator>();
```

## 获取用户详细信息

```

public static IEnumerable<IdentityResource> IdentityResources =>
    new List<IdentityResource>
    {
        new IdentityResources.OpenId(),
        new IdentityResources.Profile()
    };

```

```

AllowedScopes = {
    "api1",
    IdentityServerConstants.StandardScopes.OpenId,
    IdentityServerConstants.StandardScopes.Profile
},

```

```

var userInfo = await apiClient.GetUserInfoAsync(new UserInfoRequest
{
    Token = tokenResponse.AccessToken,
    Address = disco.UserInfoEndpoint,
});

```