ASP.Net Identity Framework

[Dokumentets underrubrik]

Andreas Elffors, Joel Persson

2014

Innehållsförteckning

[What is Identity Framework 2](#_Toc402506519)

[Why Identity Framework 2](#_Toc402506520)

[How do you use Identity Framework 2](#_Toc402506521)

[Claims & Tokens 2](#_Toc402506522)

[Principals 2](#_Toc402506523)

## What is Identity Framework

Identity framework is a framework which is designed to handle users in a secure and simple manner. It allows you to add and customize login features to your ASP.net application which makes it easy to store and handle custom data about users. The point of Identity Framework is to allow you to use the same user information to be able to access several applications. Identity has also got support for “external”/”third party” login, such as Facebook, Google, Twitter and Microsoft accounts.

Identity Framework uses something called claims based authorization which is a common way for applications to collect information about the user. This is what allows us to use third party login to authenticate your users. This means that you give the responsibility of authenticating the users to the social network giants.

## What was before Identity Framework

ASP.NET Membership was originally designed to handle application membership requirements. It uses Forms Authentication rather than Claims Identity which the Identity Framework uses. Relative to Identity Framework, ASP.NET Membership is considered “clunky” and is not very dynamic in comparison. Microsoft then introduced Simple Membership which aimed to make it easy to add membership functionally to web page applications but it still shared the same problems and disabilities as its predecessor. Universal providers is built on ASP.NET Membership and therefor shares the same limitations and deficiencies. Then the Identity Framework was introduced which implemented the OWIN framework to increase the security and it is much more dynamic in general.

## How does Identity Framework work

The Identity Framework uses claims in order to allow you to easily handle authorization on your application since the users access can be altered on a set of claims rather than a single role. This means that the application becomes more administrator friendly. The Identity Framework also uses a framework called OWIN which stands for "Open Web Interface for .Net" which is a framework that wants to decouple the server and application in order to make it easier for developers. Then we have two frameworks called OAuth and OpenId which are used to authorize and authenticate the user using a third party login.

## Claims, Tokens & Principals

Claims-based Identity is a common praxis in setting up applications to acquire the whole identity information that they would eventually need. A claim is a statement about one subject, for example a person, makes about himself or another subject. The statement can be about a name, privilege, association etc. The subject in question making the claim is the provider. And the claims are packaged into tokens witch then are issued by an issuer (provider).

An example in reality could be the authentication protocol you follow each time when in an airport. Before you go to the gate you have to check in. You presents your credentials, and your personal ID is checked (authentication) and the ticket is inspected (authorization). If found valid you get a boarding pass, and gate agents will then know your name (authentication), your flight number (authorization) etc. There are also a bar code with a serial number on your boarding pass on the back witch proves that it is issued by the airline and is not at forgery. Essentially the boarding pass is a set of claims made by the airline-company about you. With it gate agents can validate it, read claims on it and let you board the plane.

With claim-aware applications there are the ability to use an external identity system which is configured to give your application all the identity-data it needs about the user. With this approach your application no longer have to take the responsibility about authenticating users, storing user accounts and passwords etc.

The benefits of using claims based authentication is that you have the option to use third party/external login (STS – “Security Token Service”) which delegates the responsibility of authenticating the user to an STS. An example of such an STS is Facebook. Another benefit is that the users authorization can be altered by a set of claims rather than a single role, making it more flexible.

A principal represents an identity and its roles. An application can use these principals to implement role based identity.

## Important classes, interfaces & methods

### CookieAuthenticationExtensions

Has a method called UseCookieAuthentication which adds a cookie-based authentication middleware to your web application which returns an IAppBuilder (Which is a member of OWIN).

### AppBuilderExtensions

Has a method called UseExternalSignInCookie which configures the app to use OWIN middleware based cookie authentication for external identities.

### Request.GetOwinContext().Authentication.SignIn(Your Claims Identity Goes Here)

This will create a claims token (Cookie) for the recipient of the response (The User).

### Request.GetOwinContext().Authentication.SignOut(Cookietype goes here)

Removes the cookie.

### IdentityUser

Is used to create a new Identity and contains the properties to do so.

### RoleManager & UserManager

Is used when you want to create,read,update or delete roles and users from the DB.

## FAQ

### What’s the difference between claims based and roles based Identity?

Claims based identity uses one or several claims to determine your authorization while roles based identity uses the identities roles to determine the authorization.