ASP.NET Identity

ASP.NET Identity

Authorization in MVC 5 using ASP.NET Identity

ASP.NET Identity

Authorization attribute

```
[Authorize]
```

The authorize attribute can easily be applied to any controller or action which you only want authenticated identities to access.

ASP.NET Identity

Authorization attribute

The authorization attribute can also be used to check if an identity is in a specific role or is a user

```
[Authorize (Roles = "Administrator")]
```

ASP.NET Identity

Claims Authorization attribute

Since there is no attribute which can authorize an identity by its claims we will have to create our own custom “ClaimsAuthorizeAttribute”

ASP.NET Identity

We will start off by creating a class for our attribute which inherits the AuthorizeAttribute. As well as creating a constructor for the attribute which takes the claims type and value

```
public class ClaimsAuthorizeAttribute : AuthorizeAttribute
{
     private string claimType;
     private string claimValue;
     public ClaimsAuthorizeAttribute(string type, string value)
     {
           this.claimType = type;
           this.claimValue = value;
     }
 }
```

ASP.NET Identity

Next we override the OnAuthorization method to make it check if the user has a certain claim.

```
public override void OnAuthorization(AuthorizationContext filterContext)
       {
           var user = HttpContext.Current.User as ClaimsPrincipal;
           if (user.HasClaim(claimType, claimValue))
           {
               base.OnAuthorization(filterContext);
           }
           else
           {
               base.HandleUnauthorizedRequest(filterContext);
           }
       }
```

ASP.NET Identity

Now we can apply our custom attribute just like we did with the Authorize attribute.

```
[ClaimsAuthorize("Role", "Administrator")]
```

This attribute will now check if the identity has a claim of the **type** Role and **value** Administrator