Working with Claims in Our Web Application

Kevin Dockx
ARCHITECT

@KevinDockx https://www.kevindockx.com

Coming Up

Claims Transformation

Calling the UserInfo Endpoint

Role-based Authorization

Demo

Claims Transformation: Keeping the Original Claim Types

Demo

Claims Transformation: Manipulating the Claims Collection

Getting Additional Information Through the UserInfo Endpoint

We can manually call the UserInfo endpoint

- Keeps the authentication cookie small
- Allows us to get the most up-to-date information on the user

```
GET idphostaddress/connect/userinfo
Authorization: Bearer R9aty5OPlk
```

UserInfo Request

GET, but POST is also supported

Access token as Bearer token in the Authorization

Demo

Getting Ready for Calling the UserInfo Endpoint

Demo

Manually Calling the UserInfo Endpoint to Get More Claims

Role-based Authorization

Authentication

- The process of determining who you are

Authorization

- The process of determining what you are allowed to do
  - RBAC: role-based access control
  - ABAC: attribute-based access control

Demo

Role-based Authorization: Ensuring the Role Is Included

Demo

Role-based Authorization: Using the Role in Our Views

Demo

Role-based Authorization: Using the Role in Our Controllers

Demo

Creating an Access Denied Page

Summary

Resetting the claim mapping dictionary ensures the original claim types are kept

We can manipulate the ClaimsIdentity through ClaimActions to ensure claims are added or removed

Summary

The UserInfo endpoint can be called when additional claims are required

ASP.NET Core has built-in support for role-based authorization