

Entwicklerdokumentation

- 1 Azure Active Directory
- 2 Anbindung von AAD/SSO für eine normale ASP.NET Anwendung
 - 2.1 Authentifizierung
 - 2.2 Gewährung der Zugriffsrechte auf die Gruppeninformationen im AAD
 - 2.3 Verwendung in Controllern/ASP-Seiten
 - 2.4 Authorisierung über AAD-Gruppen
- 3 Anpassung der Login-Seiten

Damit man die Authentifizierung einer Webanwendung über Microsoft's Azure Active Directory abwickeln lassen kann, muss sowohl auf der Azure-Seite als auch anwendungseitig einiges konfiguriert werden. Dabei soll dieses Dokument helfen.

Azure Active Directory

Man benötigt (natürlich) einen Microsoft Account. Ein Microsoft Live/Outlook-Account funktioniert auch, so dass man auch als Entwickler gut entwickeln und testen kann, ohne Produktiv-Instanzen zu beeinflussen oder administrieren zu müssen.

1. Zunächst muss man über das Azure-Portal ein Azure Active Directory (aka Mandant) erstellen:

The screenshot shows the Azure portal interface. At the top, there is a blue header bar with the Microsoft Azure logo and a search bar labeled "Nach Ressourcen". Below the header, the URL "Home > Neu > Marketplace > Azure Active Directory >" is visible. The main content area has a title "Mandanten erstellen". There are two input fields: "Organisationsname *" containing "schneide-demo" and "Name der Anfangsdomäne *" containing "schneidedemo". Below these fields, the domain "schneidedemo.onmicrosoft.com" is shown. A dropdown menu for "Land oder Region" is open, showing "Deutschland". At the bottom of the form, a green success message says "Klicken Sie hier, um Ihren neuen Mandanten zu verwalten." (Click here to manage your new tenant).

2. Dann zumindest einen Testnutzer erstellen:

The screenshot shows the Azure portal interface under the "schneide-demo" tenant. The left sidebar has sections like "Alle Benutzer (Vorschau)", "Gelöschte Benutzer (Vorschau)", "Zurücksetzen des Kennworts", "Benutzeroberstellungen", "Diagnose und Problembehandlung", "Aktivität", "Anmeldungen", "Überwachungsprotokolle", and "Ergebnisse von Massenvorgängen". The main content area shows a table of users. The table has columns: "Name", "Benutzerprinzipalname", "Benutzertyp", "Verzeichnis synchronisiert", and "Identitätsaussteller". Two users are listed: "mihael.koep@outlook.de Koep" (Principal name: mihael.koep_outlook.de#EXT#@schneide..., Type: Member, Synchronization: No, Identifier: schneidedemo.onmicrosoft.com) and "Vae Victis" (Principal name: victus@schneidedemo.onmicrosoft.com, Type: Member, Synchronization: No, Identifier: schneidedemo.onmicrosoft.com). A note at the top of the table says "Diese Seite enthält Vorschauversionen, die Ihnen zum Testen zur Verfügung stehen. Vorschauversionen anzeigen →".

Die Oberfläche ist an dieser Stelle recht intuitiv und spricht für sich selbst.

3. Nun sollte man eine **App-Registrierung** anlegen

The screenshot shows the 'Anwendung registrieren' (Register Application) page in the Microsoft Azure portal. The 'Name' field is filled with 'OpenIDConnect-v2-Sample'. Under 'Supported Account Types', the radio button for 'Nur Konten in diesem Organisationsverzeichnis (nur "schneide-demo" – einzelner Mandant)' is selected. The 'Redirect URI (optional)' dropdown is set to 'Web' and contains the value 'https://localhost:44368/'. A note at the bottom states: 'Indem Sie den Vorgang fortsetzen, stimmen Sie den Microsoft-Plattformrichtlinien zu.' (By continuing the process, you agree to the Microsoft platform terms of service.)

Indem Sie den Vorgang fortsetzen, stimmen Sie den Microsoft-Plattformrichtlinien zu.

[Registrieren](#)

Der Name kann frei gewählt werden, sollte aber leicht in Bezug mit der entsprechenden Anwendung gebracht werden können. In unseren Anwendungsfällen möchte man nur Konten aus dem entsprechenden Organisationsverzeichnis (aka Mandant/Tenant) zulassen. Die Umleitungs-URI ist dann die URL auf die der Nutzer nach erfolgreichem Login weitergeleitet wird und damit typischerweise die Adresse der eigenen Anwendung.

In der Entwicklung

In der Entwicklung ist die Umleitungs-URI normalerweise eine Adresse auf `localhost`. Die URL sollte das HTTPS-Protokoll unterstützen, verwendet man IIS Express muss der **Port mit 443 beginnen!**

Bei der App-Registrierung muss man sicherstellen, dass man den "Implicit Grant" ("Implizite Gewährung") für das ID-Token erlaubt.

Anbindung von AAD/SSO für eine normale ASP.NET Anwendung

Die Azure Active Directory Anbindung erfordert eine OWIN-Kompatible ASP.NET-Anwendung, da nur hier die Konfiguration bequem möglich ist. Diese Konfiguration erfolgt in der `Configuration()`-Methode `Startup`-Klasse mit den gewohnten Mitteln. Durch unsere `AzureActiveDirectoryAuth`-Bibliothek ist sehr wenig Code notwendig, um die Anbindung durchzuführen und die Authentifizierungs-Informationen abzufragen.

Authentifizierung

1. Erzeugung einer OWIN-Kompatiblen Startup-Klasse damit man die Authentifizierung konfigurieren kann. Dazu muss man `OWIN` bzw. `Microsoft.Owin` als Abhängigkeit hinzufügen.
2. Dann muss die ganze Authentifizierung konfiguriert werden. Dazu werden einige Konfigurationswerte benötigt. Da diese `Ids` und `Secrets` für das Vertrauensverhältnis zwischen der Anwendung (aka Service Provider, SP) und dem Authentifizierungsdienst (aka Tenant, auch Identity Provider, IDP) wichtig sind, sollten sie nicht kompromittiert werden und auch nicht in der Versionskontrolle landen. Um die Einstellungen von der Anwendungskonfiguration in `Web.config` zu trennen, kann man folgende Anpassung machen:

web.config

```
<configuration>
    <appSettings file="PrivateSettings.config">
        <add key="webpages:Version" value="3.0.0.0"/>
        <add key="webpages:Enabled" value="false"/>
        <add key="ClientValidationEnabled" value="true"/>
        <add key="UnobtrusiveJavaScriptEnabled" value="true"/>
    </appSettings>
    ...
</configuration>
```

Damit verweist man auf eine weitere Datei die "appSettings" enthält. Die Datei sieht dann folgendermaßen aus:

PrivateSettings.config (Multi-Tenant)

```
<?xml version="1.0" encoding="utf-8"?>
<appSettings>
    <add key="ClientId" value="aaaaaaaa-1111-2222-3333-bbbbbbbbbbbb" />
    <add key="TenantId" value="cccccccc-4444-5555-6666-ddddddddddd" />
    <!-- This is an example for a multi-tenant application, that defines another allowed tenant -->
    <add key="SecondAllowedTenantId" value="cccccccc-9999-8888-7777-ddddddd" />
    <!-- For multi-tenant applications you can replace organizations with "common" to allow personal accounts, too. -->
    <add key="Authority" value="https://login.microsoftonline.com/organizations/v2.0" />
    <add key="redirectUri" value="https://localhost:44355/" />
    <add key="ClientSecret" value="_7yQ_F2v-oPKP9.19DpzYI.GS2280kc9W9" />
</appSettings>
```

Die ClientId entnimmt man der **App-Registrierung** und den Tenant der Mandanten-ID des entsprechenden Azure Active-Directory. Beides kann man auch im Nachhinein im Azure-Portal einsehen. Das ClientSecret kann man auch in der App-Registrierung erstellen, mit einem Ablaufdatum versehen - sofern man möchte - und in diese private Konfiguration übertragen. Benötigt man nur die Authentifizierung gegen **ein** Azure AD (aka "Single-Tenant"-Anwendung) sie die Konfiguration der Authority etwas anders aus:

PrivateSettings.config (Single-Tenant)

```
<?xml version="1.0" encoding="utf-8"?>
<appSettings>
    <add key="ClientId" value="aaaaaaaa-1111-2222-3333-bbbbbbbbbbbb"
/>
    <add key="TenantId" value="cccccccc-4444-5555-6666-ddddddddddd"
/>
    <!-- Use urls like below for single tenant applications -->
    <add key="Authority" value="https://login.microsoftonline.com/{0}
/v2.0" />
    <add key="redirectUri" value="https://localhost:44355/" />
    <add key="ClientSecret" value="_7yQ_F2v-oPKP9.19DpzYI.GS2280kc9W9"
/>
</appSettings>
```

3. Durch unsere AzureActiveDirectoryAuth-Bibliothek, die einfach in den Projektabhängigkeiten referenziert werden kann, ist der notwendige Konfigurations-Code minimal:

```

public class Startup
{
    public void Configuration(IAppBuilder app)
    {
        // For multi-tenant applications populate this list
        // with all the tenants where authenticated users are
        // allowed to access this application. This
        // information can come from any source like configuration,
        // database, some other webservices etc.
        // For single-tenant applications leave the list
        empty or do not set the ValidTenants property of
        // the OpenIdSettings.
        var allowedTenants = new List<string>
        {
            System.Configuration.ConfigurationManager.
            AppSettings["SecondAllowedTenantId"],
            System.Configuration.ConfigurationManager.
            AppSettings["TenantId"],
        };
        app.ConfigureOpenIdAuthentication(new OpenIdSettings
        {
            ClientId = System.Configuration.ConfigurationManager.
            AppSettings["ClientId"],
            TenantId = System.Configuration.ConfigurationManager.
            AppSettings["TenantId"],
            Authority = System.Configuration.ConfigurationManager.
            AppSettings["Authority"],
            RedirectUri = System.Configuration.ConfigurationManager.
            AppSettings["redirectUri"],
            ClientSecret = System.Configuration.
            ConfigurationManager.AppSettings["ClientSecret"],
        });
    }
}

```

Gewährung der Zugriffsrechte auf die Gruppeninformationen im AAD

Damit die Anwendung die Gruppen und deren Namen eines Benutzers über die Graph API von Microsoft erfragen kann, benötigt sie den **Scope Group.Read.All**. Für diesen muss **einmalig** ein Administrator des AADs auf das die Anwendung Zugriff erhalten soll, diesen Zugriff erlauben. Versucht man sich einzuloggen, ohne diesen "Admin Consent", so bekommt eine entsprechende Aufforderung:



thor@schneidedemo.onmicrosoft.com

Administratorgenehmigung erforderlich

AzureSSO Admin Consent

AzureSSO Admin Consent benötigt, um auf Ressourcen in Ihrer Organisation zugreifen zu können, eine Berechtigung, die nur ein Administrator erteilen kann. Bitten Sie einen Administrator, die Berechtigung für diese App zu erteilen, damit Sie die App verwenden können.

Wenn Sie über ein Administratorkonto verfügen, melden Sie sich mit diesem an.

Zur Anwendung zurückkehren, ohne Zustimmung zu erteilen

Wenn man nun zu einem AAD-Administrator-Account wechselt, kann man die Applikation freischalten:



mihael.koep@outlook.de

Angeforderte Berechtigungen

AzureSSO Admin Consent
[App-Info](#)

Diese Anwendung wird nicht von Microsoft veröffentlicht.

Diese App benötigt folgende Berechtigungen:

- ✓ Alle Gruppen lesen
- ✓ Grundlegendes Profil von Benutzern anzeigen
- ✓ Zugriff auf Daten beibehalten, für die Sie Zugriff erteilt haben

Zustimmung im Namen Ihrer Organisation

Wenn Sie zustimmen, erhält diese App Zugriff auf die angegebenen Ressourcen für alle Benutzer in Ihrer Organisation. Niemand sonst wird zur Überprüfung dieser Berechtigungen aufgefordert.

Durch das Akzeptieren dieser Berechtigungen gestatten Sie dieser App, Ihre Daten gemäß den Vertragsbedingungen und den Datenschutzbestimmungen zu verwenden. Unter <https://myapps.microsoft.com> können Sie diese Berechtigungen ändern. [Details anzeigen](#)

Wirkt diese App verdächtig? [Hier melden](#)

[Abbrechen](#)

[Akzeptieren](#)

Ob/wie das ohne Interaktion mit der Anwendung geht, ist aktuell unbekannt.

Verwendung in Controllern/ASP-Seiten

Nachdem die Anbindung an AAD erfolgt ist, kann man mit einigen wenigen Mitteln flexibel auf den Authentifizierungsstatus des Benutzers reagieren.

Abfrage des Authentifizierungsstatus

Überall, wo es sinnvoll/notwendig ist, zu prüfen, ob der Request von einem authentifizierten Nutzer kommt, kann man das direkt über ein Property am Request-Objekt abfragen:

```
if (Request.IsAuthenticated)
{
    ...
}
```

Wenn ein Benutzer authentifiziert ist, bekommt man seine, im AAD hinterlegten Informationen über seine Identität, die auch in Controllern und ASP-Seiten automatisch verfügbar ist:

```
var userClaims = new AadInformation(User.Identity as ClaimsIdentity);

ViewBag.Name = userClaims.Name;
ViewBag.Username = userClaims.Username;
ViewBag.TenantId = userClaims.TenantId;
ViewBag.Groups = string.Join(", ", userClaims.Groups);
```

Authorisierung über AAD-Gruppen

Damit man ein einfaches Authorisierungskonzept umsetzen kann, ist ein gängiger Weg Benutzergruppen zu definieren, die unterschiedliche Berechtigungen haben. Auch hier sind Einstellungen im Azure Portal und der Webanwendung vorzunehmen:

Gruppen im Azure Portal erzeugen

Man navigiert zum entsprechenden Mandanten und wählt das "Blade" Gruppen. Dort können die Gruppen und ihre Mitglieder verwaltet werden. Gruppen sind dabei hierarchisch und können also auch Untergruppen beinhalten.

Nicht mehr notwendig

Normalerweise müsste man jetzt bei der App-Registrierung in der Tokenkonfiguration einen "Gruppenanspruch" (GroupClaim) hinzufügen, damit die Gruppen des authentifizierten Nutzers auch an die Webapplikation übertragen werden. Da dadurch jedoch nur die GUIDs der Gruppen im Token mitübertragen werden, können wir uns diesen Schritt sparen. Wir fragen in jedem Fall die MS Graph API an, um

- Alle Gruppen des Nutzers zu bekommen, nicht nur die ersten 150.
- Die Namen der Gruppen zu kommen.

Authorisierung in der Webanwendung über Gruppen realisieren

Nicht mehr notwendig

Hat man nur die Gruppen-GUIDs zur Verfügung, weil man die Graph-API nicht nutzen kann oder will, muss man die Bedeutung der Gruppen-GUIDs in der Konfiguration der Anwendung hinterlegen:

```
<appSettings>
    <add key="GroupAdmin" value="74b69a88-e7da-4061-9533-7fb22b529d16"
    />
    <add key="GroupUser" value="5b135fdd-86ee-4549-a008-5f11918c9cbd"
    />
</appSettings>
```

Da wir die für die Authorisierung die Gruppennamen verwenden, ist dies **nicht** mehr notwendig!

Nach erfolgreicher Authentifizierung hat man Zugriff auf den eingeloggten Benutzer und kann an den entsprechenden Stellen erfragen, ob er Mitglied der entsprechenden Gruppe ist:

```
if (!Request.IsAuthenticated) return View();

var user = new AadInformation(User.Identity as ClaimsIdentity);
// Gruppen des aktuellen Nutzer anzeigen
ViewBag.Groups = string.Join(", ", user.Groups);
if (user.IsMemberOf(GroupConfiguration.Admin))
{
    // Dinge, die nur Admins sehen dürfen definieren
}

return View();
```

Anpassung der Login-Seiten

Es passiert leicht, dass Anwender verunsichert sind, wenn sie von der Anwendungsseite auf Anmeldungsseiten von Microsoft umgeleitet werden, ohne dass diese einen Hinweis bieten, dass es beim Login immer noch um die gewünschte Anwendung geht. Microsofts Azure bietet deswegen Möglichkeiten, die Gestaltung der Masken mit dem Anwendungs/Unternehmensbranding zu versehen. Im Azure-Portal kann man das für jedes AD gesondert konfigurieren:

Microsoft Azure Nach Ressourcen, Diensten und Dokumenten suchen (G+)

Home > schneide-demo

Unternehmensbranding bearbeiten

Azure Active Directory

Speichern Verwerfen

Hintergrundbild für Anmeldeseite
Bildgröße: 1920 x 1080 Pixel
Dateigröße: < 300 KB
Dateityp: PNG, JPG oder JPEG

Bannerlogo
Bildgröße: 280 x 60 Pixel
Dateigröße: 10 KB
Dateityp: PNG, JPG oder JPEG (transparent)

Benutzernamenhinweis

Text für die Anmeldeseite

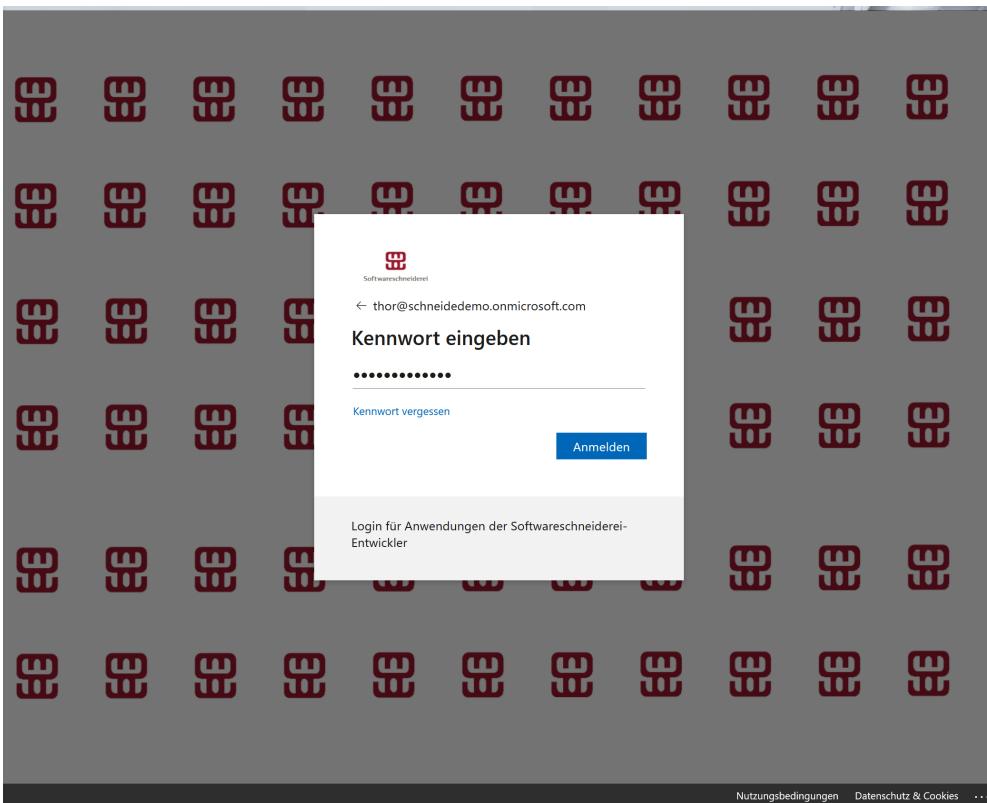
Erweiterte Einstellungen

Hintergrundfarbe der Anmeldeseite #ffffff

Bild für quadratisches Logo
Bildgröße: 240 x 240 Pixel (Größe änderbar)
Maximale Dateigröße: 50 KB
PNG (bevorzugt), JPG oder JPEG

Bild für quadratisches Logo, dunkles Design

Das führt dann zu entsprechend angepassten Anmeldeseiten, sobald die Domäne/der Tenant des Nutzers (also das AD, gegen das er sich authentifiziert) bekannt ist:



Des Weiteren ist es möglich, für jede App-Registrierung ein Branding zu hinterlegen:

Microsoft Azure michael.koep@outlook.de SCHNEIDE-DEMO

Home > schneide-demo > Graph-Multi

Graph-Multi | Branding

Suchen (STRG+ /) Speichern Verwerfen | Haben Sie Feedback für uns?

Übersicht Name * ⓘ Graph-Multi

Schnellstart Logo

Integrations-Assistent | Vorschau

Verwalten

- Branding** (circled)
- Authentifizierung
- Zertifikate & Geheimnisse
- Tokenkonfiguration
- API-Berechtigungen
- Eine API verfügbar machen
- Besitzer
- Rollen und Administratoren | Vor...
- Manifest

Support + Problembehandlung

- Problembehandlung
- Neue Supportanfrage

Branding

Name: Graph-Multi

Logo: SchneideDemo

Neues Logo hochladen: Datei auswählen

URL der Startseite: https://localhost:44355/ ✓

URL zu den Vertragsbedingungen: Beispiel: https://myapp.com/termsofservice

URL zur Datenschutzerklärung: Beispiel: https://myapp.com/privacystatement

Herausgeberdomäne: schneidedemo.onmicrosoft.com **Domäne aktualisieren**
Auf dem Einwilligungsbildschirm dieser Anwendung wird "Nicht überprüft" angezeigt.
[Weitere Informationen zur Herausgeberdomäne](#)

Herausgeberüberprüfung

MPN-ID: MPN-ID zum Überprüfen des Herausgebers hinzufügen
Die Domäne des Anwendungsherausgebers ist auf schneidedemo.onmicrosoft.com festgelegt, aber onmicrosoft.com-Herausgeberdomänen sind nicht zulässig. Verwenden Sie eine benutzerdefinierte Domäne, um den Vorgang fortzusetzen. Hinweis: Diese Domäne muss eine DNS-verifizierte Domäne auf dem Mandanten sein und mit der Domäne des primären Kontakts für Ihr MPN-Konto übereinstimmen.

Anzeigename des Herausgebers: Nicht angegeben

Wo überall die hier hinterlegten Eigenschaften angezeigt werden, ist noch ungeklärt, höchst wahrscheinlich jedoch mindestens beim gewähren der Zugriffsrechte auf Kontoinformationen für die Applikation.