# How to authorize Node.js API with Azure AD

## Introduction

This is a web app sample with Node.js, and it is demonstrates how to authorize Node.js API with Azure AD,

It’s used the adal-node library to authorize with Azure AD.

In this sample, you can test authorize Node.js API with Azure AD.

## Sample prerequisites

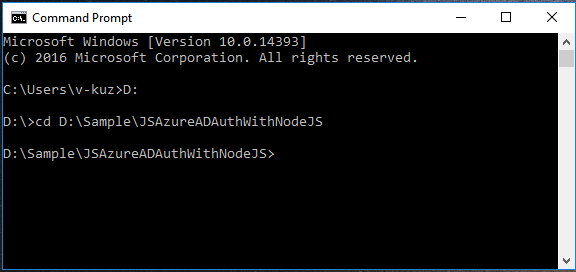
To open and run this asmple, ensure that the following requisites have been met:

* Node.js 6.6.0 or above.
* NPM is installed (In default case, when you installed Node.js, it’s had been installed).
* You have subcribed the Azure and you have the permission to manage Azure Active Directory on your subscription.

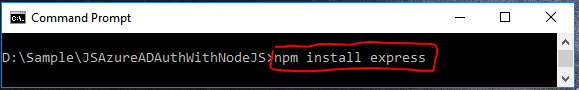
## Building the sample

**Restore the library**

* Open the Command Prompt window and navigate to sample location folder, in this case the sample location is D:\Sample\JSAzureADAuthWithNodeJS.

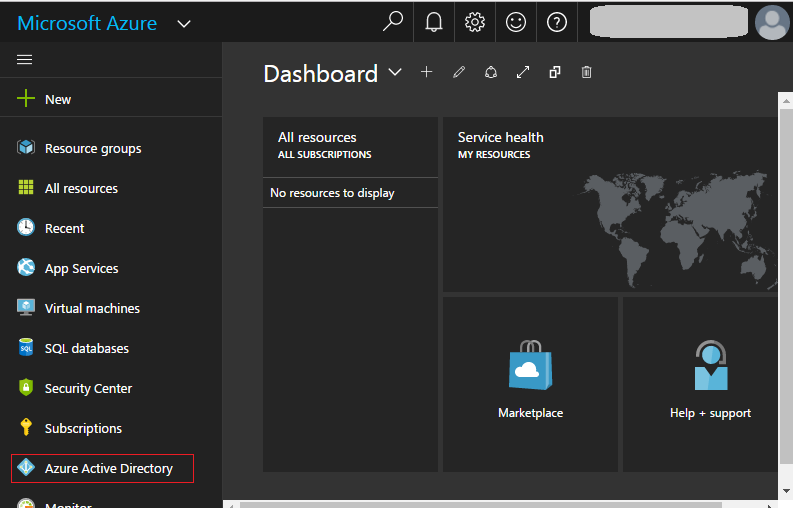


* Install the librarys
  + **npm install express**
  + **npm install cookie-parser**
  + **npm install cookie-session**
  + **npm install crypto**
  + **npm install adal-node**

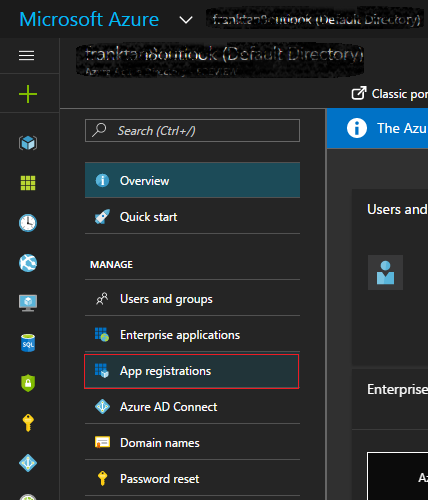


**Configuration App in Azure**

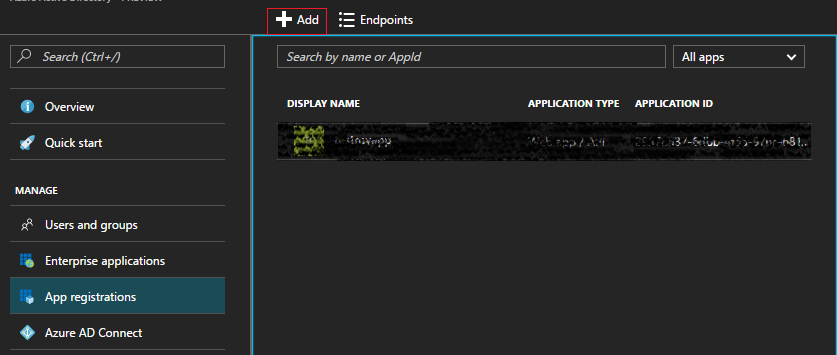
1. Go to the [**https://portal.azure.com**](https://portal.azure.com) and login.
2. In Left Panel, find the **Azure Active Directory** and click, or click the **more services** and find the **Azure Active Directory** in prop panel.



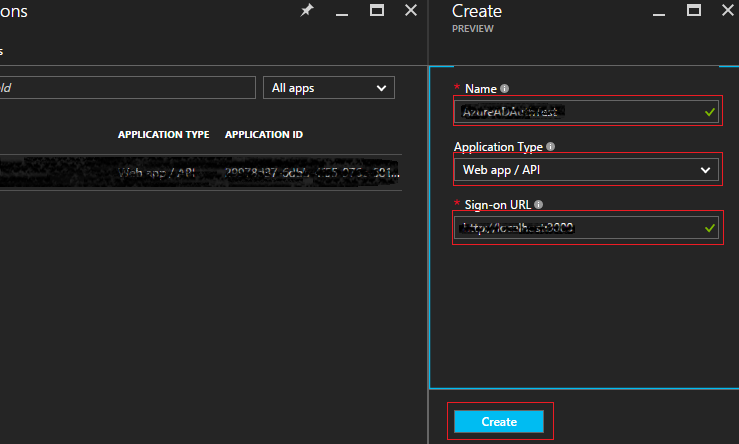
1. Click the App registrations in Azure Active Directory manage panel.



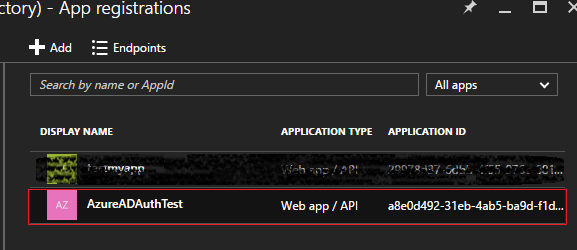
1. Click the add button.



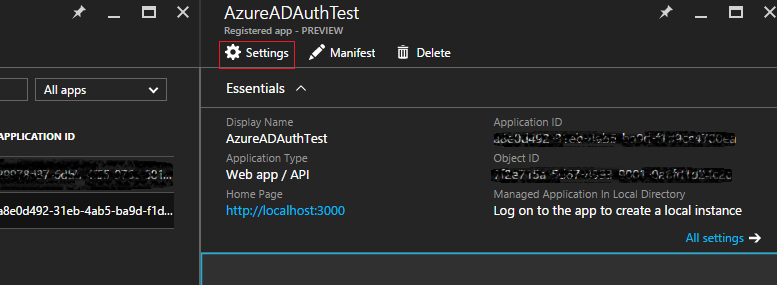
1. Fill the **Name** and **Sign-on URL**, and choose the Application Type as **Web app / API**, at last, click **Create** button.



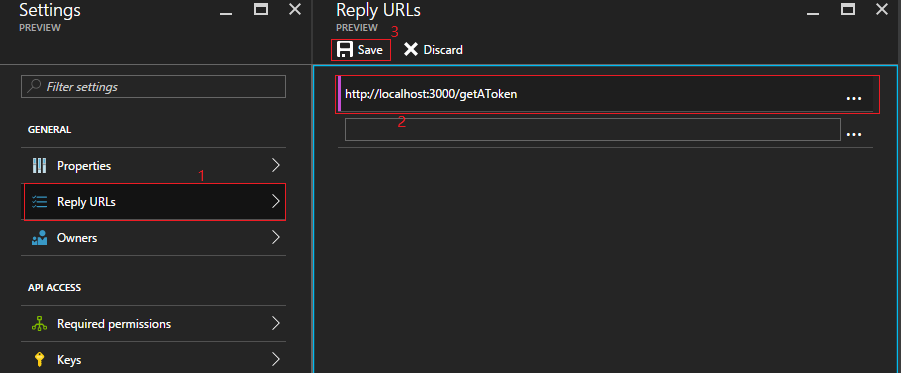
1. Then you can notice the app has been created, and show in the app list, click it to open it.



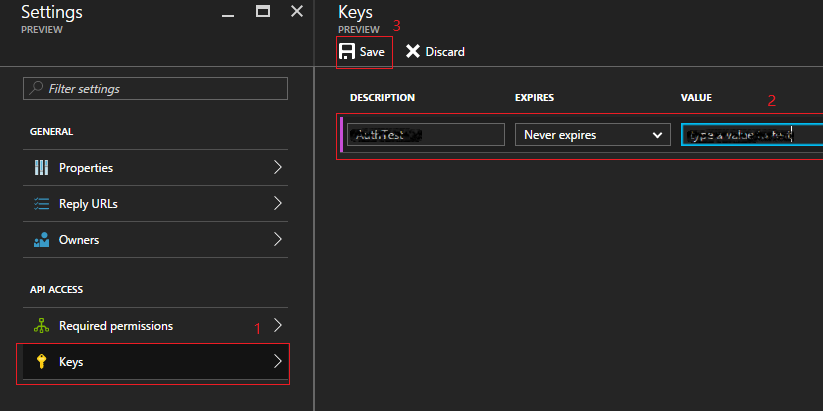
1. When you opened it, you can see under UI, click **Settings**.



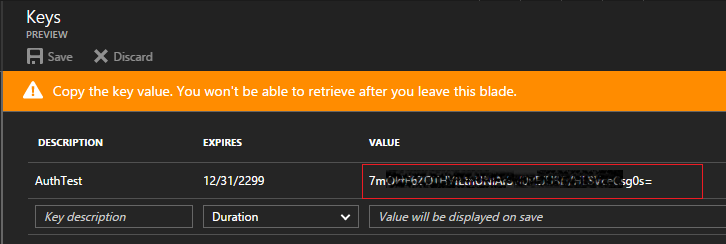
1. Click **Reply URLs**, then change address as [**http://localhost:3000/getAToken**](http://localhost:3000/getAToken), at last click the **Save** button.



1. Click Keys, then type a string in field **DESCRIPTION**, Choose EXPIRES as **Never expires**, type a string to the **VALUE** field, then click **Save** button.

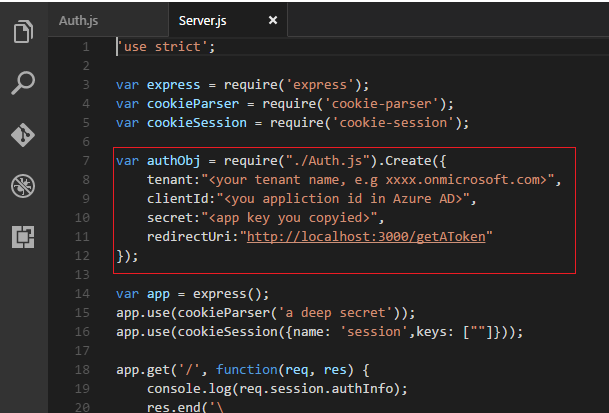


When you create success, Copy the key value, it only show once, you won’t be able to retrieve after you leave this blade.



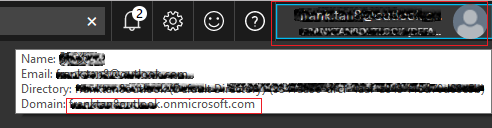
**Configuration parameters**

Open the file **server.js** and configuration flowing field.



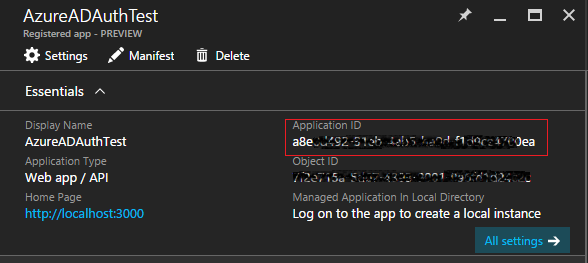
* **Tenant**

You can find it in dashboard.



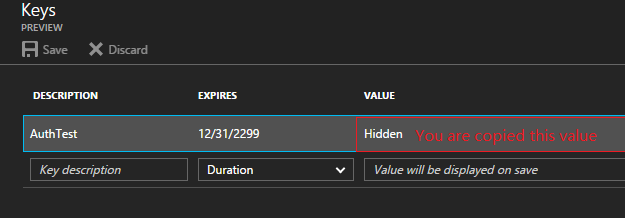
* **clientID**

**Azure Active Directory** -> **Appregistrations** -> **application of you created** -> copy the **Application ID**.



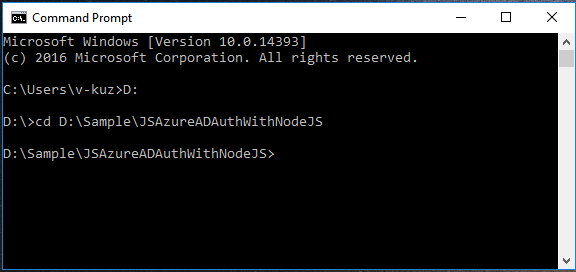
* **secret**

**application of you created** -> **Settings** -> **Keys -> you copied value.**

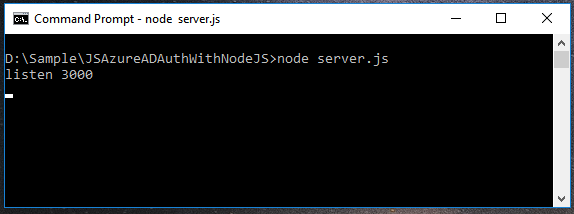


## Running the sample

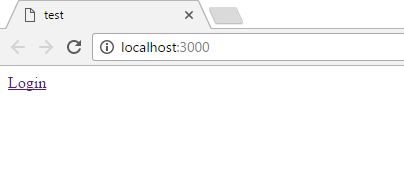
* Open the Command Prompt window and navigate to sample location folder, in this case the sample location is D:\Sample\JSAzureADAuthWithNodeJS.



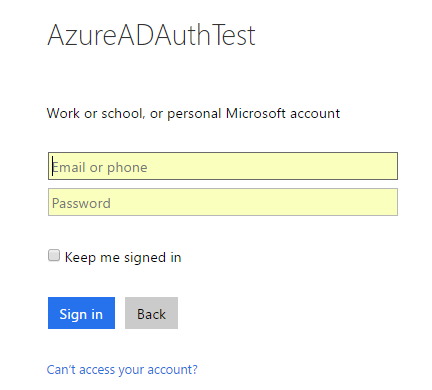
* Type command: **node server.js**, the web server will be running.



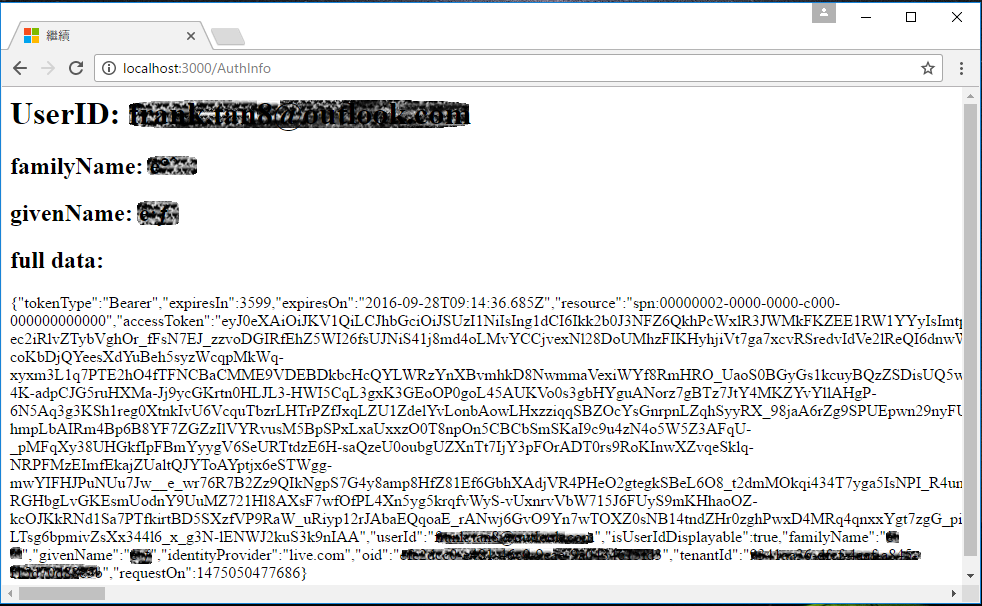
* Open the browser, and go to <http://localhost:3000>, and click the **Login** link.



* Type your Microsoft account info. Note: this account must be the member in your Azure AD.



* When authorize had finished, you can see under page, it’s contain all of authorize information.



## Using the code

**Auth.js**

'use strict';

var crypto = require('crypto');

var AuthenticationContext = require('adal-node').AuthenticationContext;

module.exports = {

Create:function(params){

var authObj = {

tenant:params.tenant,

clientId:params.clientId,

secret:params.secret,

redirectUri:params.redirectUri

};

authObj.authorityHostUrl = "https://login.windows.net";

authObj.authorityUrl = authObj.authorityHostUrl + "/" + authObj.tenant;

authObj.resource = "00000002-0000-0000-c000-000000000000";

authObj.templateAuthzUrl = 'https://login.windows.net/' + authObj.tenant + '/oauth2/authorize?response\_type=code&client\_id=<client\_id>&redirect\_uri=<redirect\_uri>&state=<state>&resource=<resource>';

authObj.loginIfNotAuth = function(req,res,action){

if(isAuthored(req))

{

if(isExpire(req))

{

authObj.refreshToken(req,res,action);

}

else{

action();

}

}

else

{

authWithAzureAD(res);

}

};

authObj.receiveToken = function(req,res,action){

if (req.cookies.authstate !== req.query.state) {

res.send('error: state does not match');

return;

}

var authenticationContext = new AuthenticationContext(authObj.authorityUrl);

authenticationContext.acquireTokenWithAuthorizationCode(req.query.code, authObj.redirectUri, authObj.resource, authObj.clientId, authObj.secret, function(err, response) {

var message = '';

if (err) {

message = 'error: ' + err.message;

res.send(message);

return;

}

response.requestOn = Date.now();

//set token to session

req.session.authInfo = response;

//do the action

if(action){

action();

}

});

};

authObj.refreshToken = function(req,res,action) {

var authenticationContext = new AuthenticationContext(authObj.authorityUrl);

authenticationContext.acquireTokenWithRefreshToken(req.session.authInfo.refreshToken, authObj.clientId, authObj.secret, authObj.resource, function(refreshErr, refreshResponse) {

if (refreshErr) {

var message = 'refreshError: ' + refreshErr.message;

res.send(message);

return;

}

refreshResponse.requestOn = Date.now();

//set token to session

req.session.authInfo = refreshResponse;

//do the action

if(action){

action();

}

});

};

function authWithAzureAD(res){

crypto.randomBytes(48, function(ex, buf) {

var token = buf.toString('base64').replace(/\//g,'\_').replace(/\+/g,'-');

res.cookie('authstate', token);

var authorizationUrl = createAuthorizationUrl(token);

res.redirect(authorizationUrl);

});

}

function isAuthored(req){

return req.session.authInfo;

}

function isExpire(req){

var now = Date.now();

var requestOn = req.session.authInfo.requestOn;

var expiresIn = req.session.authInfo.expiresIn \* 1000;

return requestOn + expiresIn >= Date.now();

}

function createAuthorizationUrl(state) {

var authorizationUrl = authObj.templateAuthzUrl.replace('<client\_id>', authObj.clientId)

.replace('<redirect\_uri>',authObj.redirectUri)

.replace('<state>', state)

.replace('<resource>', authObj.resource);

return authorizationUrl;

}

return authObj;

}

};

**Server.js**

'use strict';

var express = require('express');

var cookieParser = require('cookie-parser');

var cookieSession = require('cookie-session');

var authObj = require("./Auth.js").Create({

tenant:"<your tenant name, e.g xxxx.onmicrosoft.com>",

clientId:"<you appliction id in Azure AD>",

secret:"<app key you copyied>",

redirectUri:"http://localhost:3000/getAToken"

});

var app = express();

app.use(cookieParser('a deep secret'));

app.use(cookieSession({name: 'session',keys: [""]}));

app.get('/', function(req, res) {

res.end('\

<head>\

<title>test</title>\

</head>\

<body>\

<a href="./auth">Login</a>\

</body>\

');

});

app.get('/auth', function(req, res) {

authObj.loginIfNotAuth(req,res,function(){

res.send("authed");

});

});

app.get('/getAToken', function(req, res) {

authObj.receiveToken(req,res,function(){

res.redirect('/AuthInfo');

});

});

app.get('/AuthInfo', function(req, res) {

var sessionValue = req.session.authInfo;

var authString = JSON.stringify(sessionValue);

var userID = sessionValue.userId;

var familyName = sessionValue.familyName;

var givenName = sessionValue.givenName;

res.end(`\

<h1>UserID: ${userID}</h1>

<h2>familyName: ${familyName}</h2>

<h2>givenName: ${givenName}</h2>

<h2>full data:</h2>

<p>${authString}</p>

`);

});

app.listen(3000);

console.log("listen 3000");

## More information

Windows Azure Active Directory Authentication Library (ADAL) for Node.js

<https://github.com/AzureAD/azure-activedirectory-library-for-nodejs>