Find your information

Synology publishes a wide range of supporting documentation.

In Knowledge Center, you will find useful Help and FAQ articles, as well as video tutorials breaking up processes into handy steps. You can also find User’s Guides, Solution Guides, brochures, and White Papers. Experienced users and administrators will find answers and guidance in technical Administrator’s Guides and Developer Guides.

Got a problem and unable to find the solution in our official documentation? Search hundreds of answers by users and support staff in Synology Community or reach Synology Support through the web form, email or telephone.
Synology DSM SSO Server is based on the OAuth 2 protocol. We provide the JavaScript SDK for 3rd party development. SSO Server JavaScript SDK script will be installed automatically after SSO Server installation.
## DSM JavaScript SDK Script Location

http://DSM_IP_OR_HOSTNAME:5000/webman/sso/synoSSO-1.0.0.js

## Usage

### Initialization

SYNOSSO.init

SYNOSSO.init is used to initialize SYNOSSO SDK. You need to call SYNOSSO.init before calling any other SYNOSSO APIs.

### Function parameters of SYNOSSO.init:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oauthserver_url</td>
<td>string</td>
<td>The URL of the DSM where SSO Server is installed.</td>
</tr>
<tr>
<td>app_id</td>
<td>string</td>
<td>APP ID registered on the DSM SSO Server</td>
</tr>
<tr>
<td>redirect_uri</td>
<td>string</td>
<td>Redirect URI registered on the DSM SSO Server.</td>
</tr>
<tr>
<td>callback</td>
<td>Javascript function object</td>
<td>User defined callback for handling login query/login response.</td>
</tr>
<tr>
<td>domain_name(optional)</td>
<td>string</td>
<td>Windows AD domain name of SSO client. Ex: &quot;MYDOMAIN.COM&quot;</td>
</tr>
<tr>
<td>ldap_baseDN(optional)</td>
<td>string</td>
<td>LDAP baseDN of SSO client. Ex: &quot;dc=myldap,dc=com&quot;</td>
</tr>
</tbody>
</table>

### Notes:
- Directory service related options are for directory service checking. If one of these options is provided, SSO Server will validate if this directory service is the same as the DSM that the SSO Server belongs to.
Chapter 2: Javascript SDK

Example:

SYNOSSO.init(
  app_id: '153fcb35b01571b49cb0adca3a4bda40',
  redirect_uri: 'http://10.13.20.130:5000', // redirect url have to be the same as the one registered in SSO server, and can be a plain text html file.
  callback: authCallback
);  

Authentication

SYNOSSO.login();

After calling SYNOSSO.login, a login popup window containing a dialog for SSO will appear. SYNOSSO.login has no arguments and will call the callback registered in SYNOSSO.init after the user logs in successfully.

Example:

SYNOSSO.login();

Response:

Response of callback registered in SYNOSSO.init:

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>String: &quot;login&quot;/&quot;not_login&quot;/ERR_STRING</td>
<td>Show status of this user on SSO Server.</td>
</tr>
<tr>
<td>Access_token</td>
<td>string</td>
<td>Access token returned from SSO Server after this user logs in successfully.</td>
</tr>
</tbody>
</table>

If the user already signed in to SSO Server:

response:{
  status: 'login',
  access_token: 'ABCDE'
}

If the user hasn't signed in to SSO Server:

response:{
  status: 'not_login'
}
If any unexpected error occurs:

```javascript
response:{
    status: 'ERR_STRING'
}
```

**Notes:**
- For ERR_STRING, please refer to Chapter 6 for more details.

**Logout**

```javascript
SYNOSSO.logout(function(){
    //do something after logout.
});
```

**Function parameters of SSOSYNO.logout:**

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>callback</td>
<td>Javascript function</td>
<td>The callback which will be called after the user logs out from SSO Server.</td>
</tr>
</tbody>
</table>

**SYNOSSO.logout** has a callback which will be called after the user signs out of SSO Server.

- Before a user signs out of your application, call **SYNOSSO.logout**, and it will sign out this user from SSO Server.
- **SYNOSSO.init** must be called before **SYNOSSO.logout**.
- **SYNOSSO.logout** only signs out the user from SSO Server and will not affect the login status of users in other applications.

Response of callback of **SYNOSSO.logout** has no arguments.
Chapter 3: Manual Flow

1. Bring the user to `http://[DSM Oauth Server:5000]/webman/sso/SSOOauth.cgi` with the following query string parameters:
   - `app_id`: APP ID registered on DSM SSO Server
   - `redirect_uri`: Redirect URI registered on DSM SSO Server
   - `synossoJSSDK`: False, represents the manual flow
   - `scope`: SSO server only provides "user_id" scope, which means there is limited user information for Single-Sign On
   - `state` (optional): Used to protect CSRF

   Then, the login window will appear for the user to input their username and password.

   **Example:**
   ```
   SSO Server: 10.13.20.254  
   SSO Client: 10.13.22.128  
   ```

2. User signs in to SSO Server.

3. After signing in successfully, the user will be redirected back to the redirect URI that this app registered on SSO Server with following hash values:
   - `access_token`: The access token which will be used to exchange user information.
   - `State` (optional): If you provide the state in step 1, the exact same state will be returned.

   **Example:**
   ```
   http://10.13.22.128:5000#access_token=58322f3eaaG7t69030edH2bcdee08brWc6250eba&state=fabc21cf
   ```
Chapter 4: Exchange User Information

Exchange for user’s information

1. You need to use an access token to get the user_id and user_name.

2. Go to endpoint: `http://[DSMOauthServer:5000]/webman/sso/SSOAccessToken.cgi`
   with these query string parameters:
   - `action`: “exchange”
   - `access_token`: “ABCDE”
   - `app_id`: “asfsdfsdf3e”

Example:

```
curl http://[DSMOauthServer:5000]/webman/sso/SSOAccessToken.cgi?action="exchange"&access_token ="ABCDE"&app_id="asfsdfsdf3e"
```

Response:

If the token is correct:

```
{
success: true,

data:{
user_id:1024,
user_name:john
}
}
```

If any unexpected errors occurred:

```
{
success: false,
error: 'ERR_STRING'
}
```
Chapter 5: Example Code

Javascript SDK Examples

Fontpage.html

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Test App 1</title>
</head>
<body>
    <h1>Test App 1</h1>
    <h2>Please sign in via Synology SSO</h2>
    <button id="login-button">SSO Login</button>
    <h3 id="my-text"></h3>
</body>
<script type="text/javascript" src="https://sso.server/webman/sso/synoSSO-1.0.0.js"></script>
<script>
document.addEventListener("DOMContentLoaded", function() {
    SYNOSSO.init({
        oauthserver_url: 'https://sso.server/',
        app_id: '34744cd97de19fec4272cb99d13aca75',
        redirect_uri: 'https://10.17.188.213/',
        callback: authCallback
    });

    function authCallback(response) {
        if('not_login' === response.status) { //user not
            console.log(response.status);
        }
    }
});
```
```
```javascript
} else if('login' === response.status) {
  var xhr = new XMLHttpRequest();
  xhr.open('GET', '/login_backend.php?accesstoken=' + response.access_token);
  xhr.onload = function() {
    if (xhr.status === 200) {
      var message = ''; 
      var response = JSON.parse(xhr.responseText);
      if (response && response.success && response.data) {
        message = 'Login success, User Name= ' + response.data.user_name + ', User ID= ' + response.data.user_id;
      } else {
        message = 'Login failed. Response: ' + xhr.responseText;
      }
      document.getElementById("my-text").innerHTML = message;
    } else {
      alert('Request failed. Returned status of ' + xhr.status);
    }
  }
  xhr.onerror = function() {
    alert('Request failed due to network error.
  }
  xhr.send();

} else {
  // iframe query login status not support any more
  console.log(response.status);
}

var login_button = document.getElementById("login-button");
login_button.addEventListener('click', SYNOSSO.login);
```
Chapter 5: Example Code

Login_backend.php

```php
<?php
$accesstoken = $_GET['accesstoken'];
function httpGet ($url)
{
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $url);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_HEADER, false);
    curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false);//for testing ignore checking CA
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
    $output = curl_exec($ch);
    curl_close($ch);
    return $output;
}
$url_str = "https://aso.syno/webman/aso/SSOAccessToken.cgi?action=exchange&access_token=".$accesstoken;
header('Content-Type: application/json');
echo httpGet($url_str);
?>
```
Chapter 6: Error Strings

ERR_STRING

• server_error - SSO server error.
• parameter_error - Parameter error when SYNOSSO.init.
• invalid_app_id - APP_ID error.
• invalid_redirect_uri - Redirect URI error.
• invalid_directory_service - Different directory service between SYNOSSO.init and DSM SSO Server.
• invalid_token - Invalid SSO access token.
• unknown_error - Other unexpected errors.