Contents:

auth-sdk

The auth-sdk is just a simple wrapper around persistent state storage and redirect response parsing for the OAuth2 web grant type (http://tools.ietf.org/html/rfc6749#section-4.1).

A simpler explanation is given http://aaronparecki.com/articles/2012/07/29/1/oauth2-simplified (http://aaronparecki.com/articles/2012/07/29/1/oauth2-simplified). You should read it at least once, to have a basic understanding of OAuth2. This sdk covers the "Web Server Apps" part mentioned there.

The auth-sdk hides most of the OAuth2 stuff, so you'll basically have to do 3 things (in this order):

- First try to parse the redirect sent back to your site after the login.
    - Call sdk->parseRedirectResponse()
- Check, if you have permissions (=access token) to call an api method for the user.
    - Call sdk->getUser()
- If not, redirect the user to an external login page. [1]
    - Redirect to sdk->getLoginUrl()

Your app now should have permissions to make an api call on behalf of the user. Take a look at: `./examples/basic.php`

You can use the acces token then stored in the auth-sdk for subsequent api calls until the token expires. (Server-side or until the user logs out)

Include with composer

```
  "repositories": [
    {
    "type": "git",
    "url": "https://app-developers-89:98ashUZsujna!isi.asU7@antevorte.codebasehq.com/public-sdks-2/php-auth-sdk.git"
    },
    {
    "type": "git",
    "url": "https://app-developers-89:98ashUZsujna!isi.asU7@antevorte.codebasehq.com/public-sdks-2/php-jws.git"
    }
    ],
    "require": {
    "collins/php-auth-sdk": "0.3.4"
    }
```

If you just want to run the examples now, please refer to the "Examples" section.

Oauth2 web grant type usage

- Have a look at: `./examples/basic.php`

Create the sdk object

```
$authSDK = new AuthSDK(array(
    'clientId'=>'from_dev_center',
    'clientToken'=>'from_dev_center',
    'clientSecret' => 'from_dev_center',
    'redirectUri'=>'entered_in_dev_center',
    'scope'=>'firstname',  //optional. the only valid value is firstname at the moment
    'popup'=>'true|false', //optional. true will render a small login, false the login webpage
    ),new SessionStorage());
```

(There are two optional parameters that can be "overwritten" 'loginUrl' and 'resourceUrl'. But usually there is no need for that)

Parse the response (login redirected back to your site)

- Call this method first and only once per request
- $parsed will be true, if it was a redirect. Just in case you need this information.

```
$parsed = $authSDK->parseRedirectResponse();
```

Check user status. Show login or logout

- Check, if login button|redirect needed

```
$authResult = $authSDK->getUser();
    if($authResult->hasErrors()){
    //optional, add values you want to get back on your redirect endpoint
    //but do this before getLoginUrl()
    $authSDK->setState('someKey','someVal');

    renderLoginButton( $authSDK->getLoginUrl() ); //renderLoginButton: your render method
    var_dump($authResult->getErrors());

    }else{
    renderLogoutButton( $authSDK->getLogoutUrl() ); //renderLogoutButton: your render method
    var_dump($authResult->getResult()->response);
    var_dump($authSDK->getState('someKey'));

    }
```

- Its also possible to set 'state' params (will be returned)
- getLogoutUrl() has an optional parameter redirectUrl. If set, after the logout you will be redirected to that. Else to the 'redirectUri' of the sdk config.
- In both cases getLogoutUrl() will append the get parameter logout=true to redirectUrl (which is used by e.g. parseRedirectUrl() to clean the persistent storage).

Examples

- Can be found in the folder: `./examples/*`

Running

- Open a terminal, in the sdk root folder run `php composer.phar install`.

- If you use php >= 5.4 you can now try `./run.sh` (as root/sudo) in the `./examples` folder. Keep an eye on the console messages (it will complain about the /etc/hosts config, add it). If everything works, you should be able to open the (local) url shown in the console afterwards.

- On Windows, or if run.sh is not working: With php >= 5.4 just try `php -S mytestserver.local:80` in the `./examples` folder (as root/sudo on linux/osx). Notice you have to add `127.0.0.1 mytestserver.local` to your `/etc/hosts` for this to work. (Windows: system32\drivers\etc\hosts)

- For php < 5.4 refer to your server documentation to create a vhost config with the document root pointing to the `./examples` folder.

Files

- `./examples/basic.php`:
  - example for typical usecase, just as much code as needed. Note, that you need to set your redirectUrl to {yourdomain}/basic.php
- `./examples/parent_page.php` & `./examples/result_page.php`:
  - example for showing more options, intercepts the redirect, so you can take a look at the params returned. Note, that you need to set your redirectUrl to {yourdomain}/result_page.php

Config

- Copy `./example/common_params.php` to `./example/common_params.local.php`
- Change the params in ./example/common_params.local.php to match your values (from dev center)

1.
   - If the user is not logged in there, it will grant the user for its username and password and then redirect back to your site with an access token.
   - If your user however already is logged in, it will just redirect back to your site with an access token.
   - There is one more authorization flow step after those possible grants from the user and really fetching the access token, but the auth-sdk will gently hide that from you

   ↩