API olx.com.br

Utilizando o protocolo OAuth 2.0

Resumo

Este documento descreve como utilizar o protocolo oAuth 2.0 como forma de autenticação na API olx.com.br através de uma aplicação *web*.

OAuth 2.0 é um protocolo relativamente simples. No início, você registra sua aplicação no olx.com.br, depois a aplicação solicita uma chave de acesso ao servidor de autenticação do olx.com.br e então utiliza essa chave para receber as informações de um recurso da API olx.com.br que deseja acessar.

Índice

1. Registro da aplicação

Antes de iniciar o protocolo de autenticação com o servidor olx.com.br, o cliente deverá registrar sua aplicação, fornecendo os seguintes dados:

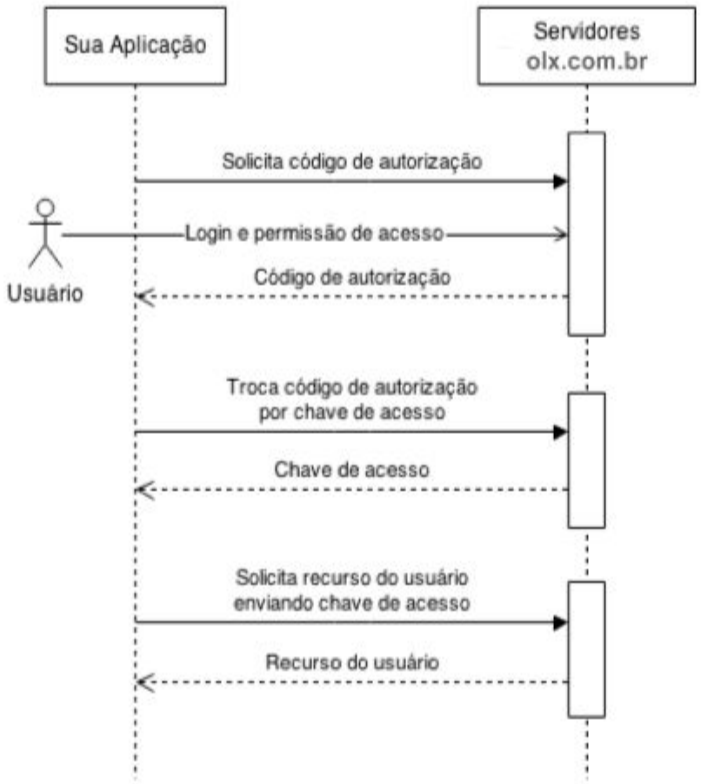
- Nome do cliente
- Nome da aplicação
- Descrição da aplicação
- Website
- Telefone
- E-mail
- URIs de redirecionamento (identifica um *end-point* do cliente que será alvo de redirecionamento no processo de autenticação; mínimo 1 e máximo 3; ver seção 2.1)

Após receber os dados, o olx.com.br entrará em contato com o cliente para fornecer sua identificação `client_id` e sua chave de segurança, necessários para iniciar a sequência de autorização.

2. Sequência de autorização

Após o cliente registrar sua aplicação no olx.com.br e obter sua identificação (client_id), será possível dar início a sequência de autorização.

A sequência de autorização tem início quando a aplicação redireciona o navegador para uma URL do olx.com.br. Nesta URL serão incluídos alguns parâmetros que indicam o tipo de acesso que está sendo requerido. O olx.com.br é o responsável pela autenticação do usuário e por confirmar a permissão de acesso a suas informações e recursos. Como resultado, o olx.com.br retorna para a aplicação um código de autorização. Após receber o código de autorização a aplicação poderá fornecê-lo (junto com sua identificação e senha) para obter em troca uma chave de acesso. A aplicação poderá então utilizar essa chave de acesso para acessar a API olx.com.br.



2.1. Código de autenticação

O código de autorização é obtido utilizando o servidor de autorização do olx.com.br como intermediador entre o cliente/aplicação e o usuário.

O servidor de autenticação valida a requisição para certificar que todos os parâmetros obrigatórios estão válidos e presentes. Se a requisição for válida o servidor de autenticação tenta autenticar o usuário (pedindo seu login e senha) e obtém dele uma decisão de autorização.

Invés de solicitar a autorização diretamente ao usuário, o cliente direciona-o para a página de autenticação do olx.com.br que, após autenticar o usuário com seu login e senha, redireciona o usuário de volta ao cliente junto com o código de autorização.

Este código deverá ser utilizado logo em seguida para solicitar a chave de acesso que, caso seja aceita pelo usuário, permitirá o cliente a acessar suas informações através da API olx.com.br. O código de autenticação é temporário e não pode ser reutilizado.

2.1.1.

A URL usada para autenticar um usuário será `https://auth.olx.com.br/oauth`

Os parâmetros HTTP "GET" suportados pelo servidor de autenticação do olx.com.br para aplicações Web são:

Parâmetro	**Valores**	**Descrição**
`response_type` (obrigatório)	`code`	Determina que o valor esperado pela requisição será um código de autorização.
`client_id` (obrigatório)	A identificação do cliente que foi fornecida pelo olx.com.br através do registro da aplicação.	Identifica o cliente que está enviando a requisição. O valor do parâmetro tem que ser idêntico ao valor fornecido pelo olx.com.br durante o registro da aplicação.
`redirect_uri` (obrigatório se múltiplos URI de redirecionamento foram cadastrados no registro na aplicação)	O valor do parâmetro tem que ser idêntico a um dos URI cadastrados no registro da aplicação no olx.com.br.	Determina para qual servidor a resposta da requisição será enviada.
`scope` (obrigatório)	Conjunto de permissões que a aplicação solicita, separadas por espaço,	Identifica o tipo de acesso a API olx.com.br que a aplicação está requisitando. Os valores passados neste parâmetro serão os mesmos

	não importa a ordem dos valores. Ver tópico Permissões.	apresentados ao usuário na tela de solicitação de permissão.
`state`	Qualquer valor.	Fornece qualquer valor que pode ser útil a aplicação ao receber a resposta de requisição.

Um exemplo de URL é apresentado abaixo com espaços e quebras de linha para maior legibilidade.

```
https://auth.olx.com.br/oauth?scope=basic_user_info&state=/profile&redirect_uri=https:
//yourserver.com/code&response_type=code&client_id=1055d3e698d289f2af8663725127bd4b
```

2.1.1.1. Permissões

Ao solicitar a chave de acesso o cliente deve solicitar permissões através do parâmetro `scope`. Essas solicitações serão enviadas ao usuário para que ele possa permitir ou não o acesso.

O valor parâmetro `scope` é expressado como uma lista de valores (*case sensitive*) separados por espaços, não importando a ordem.

O servidor de autorização do olx.com.br pode ignorar parcial ou totalmente as permissões solicitadas pelo cliente de acordo com as políticas do servidor de autenticação ou por instrução do usuário.

Segue abaixo a lista dos possíveis valores de permissão que podem ser solicitadas ao usuário:

Valor	**Descrição**
`basic_user_info`	Permite acesso as informações básicas do usuário. Ex: nome completo e email.
`autoupload`	Permite acesso ao sistema de autouploads (Envio de anúncios de forma automática)

2.1.2. Resposta da requisição

A resposta será enviada para o `redirect_uri`, conforme especificado na URL da requisição. Se o usuário aprovar o acesso, a resposta conterá um código de autorização e o parâmetro `state` (se incluído na requisição). Se o usuário não aprovar, a resposta retornará uma mensagem de erro. Todas as respostas são retornadas ao servidor web por `query string`.

2.1.2.1. Confirmação da autorização

Se o usuário conceder a permissão solicitada o servidor de autenticação gera um código de autorização e envia-o para o cliente através dos seguintes parâmetros na URI de redirecionamento:

Parâmetro	**Valores**	**Descrição**
`code` (obrigatório)	Código de autorização gerado pelo servidor de autenticação.	Código de autorização utilizado para solicitar permissão de acesso a recursos de um usuário. Expira 10 minutos após ter sido gerado e não pode ser reutilizado.
`state` (obrigatório se esteve presente na requisição)	Mesmo valor enviado pelo cliente na requisição.	Fornece qualquer valor que pode ser útil a aplicação ao receber a resposta de requisição.

Exemplo de resposta de confirmação:

```
https://yourserver.com/code?state=/profile&code=4gP7q7W91a-oMsCeLvIaQm6bTrgtp7
```

2.1.2.2. Erros de autenticação

Se a requisição falhar devido a uma URI de redirecionamento inválida ou não enviada ou se o identificador do cliente (`client_id`) inválido ou não enviado o servidor de autenticação não irá redirecionar o usuário para o URI de redirecionamento.

Se o usuário negar acesso as permissões solicitadas ou se a requisição falhar por outros motivos o servidor de autenticação informa ao cliente qual foi o erro através de parâmetros enviados na URI de redirecionamento.

Caso a URI passada seja diferente de uma das registradas no olx.com.br, o servidor de autenticação responderá com um HTTP 400 (*Bad Request*).

Segue abaixo a lista dos parâmetros para erros de autenticação retornados pelo servidor de autenticação:

Parâmetro	**Valores**	**Descrição**
`error` (obrigatório)	`invalid_request`	A requisição tem parâmetro obrigatório faltando, inclui um parâmetro inválido ou está mal

		formatada por algum outro motivo.
	`unauthorized_client`	O cliente não está autorizado a requisitar um código de autorização usando este método.
	`access_denied`	O usuário ou servidor de autenticação negou a requisição.
	`unsupported_response_type`	O servidor de autenticação não suporta obter um código de autorização utilizando este método.
	`invalid_scope`	A permissão solicitada é inválida, desconhecida ou mal formatada.
	`temporarily_unavailable`	No momento o servidor de autenticação está indisponível para receber requisições devido a uma manutenção ou sobrecarga temporária.
	`server_error`	O servidor de autenticação encontrou uma condição inesperada que impossibilitou a finalização da requisição.
`state` (obrigatório se esteve presente na requisição)	Mesmo valor enviado pelo cliente na requisição.	Fornece qualquer valor que pode ser útil a aplicação ao receber a resposta de requisição.

Exemplo de resposta de erro:

```
https://yourserver.com/code?error=access_denied&state=/profile
```

2.2. Chave de acesso

Chaves de acesso são credenciais usadas para acessar recursos protegidos de um usuário. É uma string que representa uma autorização emitida ao cliente. Esta chave representa permissões específicas, concedidas pelo usuário e emitida pelo servidor de autorização.

2.2.1. Requisição da chave de acesso

A URL usada para solicitar a chave de acesso será `https://auth.olx.com.br/oauth/token`

Após receber o código de autorização, o cliente poderá troca-lo por uma chave de acesso através de uma nova requisição. Esta requisição deverá ser um *post* HTTPS e conter os seguintes parâmetros:

Parâmetro	**Valores**	**Descrição**
`code` (obrigatório)	O código de autorização retornado na requisição anterior.	Código de autorização utilizado para solicitar permissão de acesso a recursos de um usuário. Expira 10 minutos após ter sido gerado e não pode ser reutilizado.
`client_id` (obrigatório)	A identificação do cliente que foi fornecida pelo olx.com.br através do registro da aplicação.	Identifica o cliente que está enviando a requisição. O valor do parâmetro tem que ser idêntico ao valor fornecido pelo olx.com.br durante o registro da aplicação.
`client_secret` (obrigatório)	A chave de segurança enviada pelo olx.com.br durante o registro da aplicação.	A chave de segurança do cliente obtida no registro da aplicação no olx.com.br.
`redirect_uri` (obrigatório se enviado na requisição do código de autorização)	Mesmo valor enviado através do parâmetro `redirect_uri` na requisição do código de autorização.	Determina para qual servidor a resposta da requisição será enviada.
`grant_type` (obrigatório)	`authorization_code`	Identifica a forma como o cliente obteve autorização para solicitar a chave de acesso.

Segue abaixo um exemplo de requisição de chave de acesso:

```
POST /oauth/token HTTP/1.1
Host: auth.olx.com.br
Content-Type: application/x-www-form-urlencoded

code=4/P7q7W91a-oMsCeLvIaQm6bTrgtp7&
client_id=1055d3e698d289f2af8663725127bd4b&
client_secret={sua_chave_de_segurança}&redirect_uri=https://yourserver.com/code&
grant_type=authorization_code
```

2.2.2. Resposta de requisição

Se a solicitação da chave de acesso for válida e autorizada, o servidor de autenticação retorna um HTTP 200 (OK) com valor da chave de acesso. Se a solicitação falhar o servidor de autenticação, retorna um erro conforme seção 2.2.2.2.

2.2.2.1. Resposta bem sucedida

O servidor de autenticação gera uma chave de acesso e constrói a resposta adicionando os seguintes parâmetros no corpo da resposta HTTP 200 (*OK*):

Parâmetro	**Valores**	**Descrição**
`access_token` (obrigatório)	A chave de acesso retornada pelo servidor de autenticação.	A chave de acesso que será utilizada para utilizar a API olx.com.br.
`token_type` (obrigatório)	`Bearer`	Define o tipo de chave gerada.

Segue abaixo um exemplo de resposta bem sucedida em formato JSON retornada pelo servidor de autenticação:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache

{
  "access_token":"1/fFAGRNJru1FTz70BzhT3Zg",
  "token_type":"Bearer"
}
```

2.2.2.2. Resposta de erro

O servidor de autenticação retorna um HTTP 400 (*Bad Request*) e inclui um dos seguintes valores na resposta:

Parâmetro	**Valores**	**Descrição**
`error` (obrigatório)	`invalid_request`	Falta de parâmetro obrigatório na requisição, inclui um parâmetro não suportado, inclui um parâmetro repetido, incluir múltiplas credenciais ou mal formato por algum outro motivo.

	`invalid_client`	Falha na autorização do cliente.
	`invalid_grant`	Código de autorização inválido, expirado ou revogado, URI não é a mesma que a URI usada na requisição de autorização ou foi emitido para outro cliente.
	`unauthorized_client`	O cliente autenticado não está autorizado a utilizar este tipo de autorização (`grant_type`).
	`unsupported_grant_type`	O tipo de autorização (`grant_type`) não é suportado pelo servidor.
	`invalid_scope`	A solicitação de permissão enviada é inválido, desconhecida ou excede as permissões concedidas pelo usuário.

Segue abaixo um exemplo de resposta de erro na requisição de uma chave de acesso:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache

{
       "error":"invalid_request"
}
```

3. API olx.com.br

Depois que a aplicação obtiver uma chave de acesso poderá então utilizar a API olx.com.br para acessar recursos privados de um determinado usuário.

3.1 Acesso a API

O cliente acessa os recursos protegidos do usuário apresentando a chave de acesso a API olx.com.br. O servidor valida a chave de acesso e garante que as permissões concedidas são suficientes para acessar o recurso solicitado.

Segue abaixo um exemplo de como utilizar a API olx.com.br para obter as informações básicas de um usuário:

```
POST /oauth_api/basic_user_info HTTPS/1.1
Host: apps.olx.com.br
User-Agent: Mozilla/5.0
Content-Type: application/json;charset=UTF-8
{
  "access_token":"387266f574068c83f74942fe255e82261708f960"
}
```

Ou utilizando uma aplicação por linha de comando como, por exemplo, o CURL:

```
curl -X POST https://apps.olx.com.br/oauth_api/basic_user_info --data
'{"access_token":"387266f574068c83f74942fe255e82261708f960"}' -H 'User-Agent:
Mozilla/5.0'
```

3.2 Referência

A versão atual da API OLXfornece acesso aos seguintes recursos:

Recurso	**Descrição**
`basic_user_info`	Retorna uma lista com as informações básicas do usuário.
`autoupload`	Sistema de envio de anúncios de forma automática

3.2.1 basic_user_info

Retorna uma lista com as informações básicas do usuário.

Acesso

```
POST /oauth_api/basic_user_info HTTPS/1.1
Host: apps.olx.com.br
```

Permissões

- Qualquer chave de acesso com a permissão `basic_user_info`.

Campos

Nome	Descrição	Tipo
`user_name`	Nome completo do usuário.	`string`
`user_email`	Email do usuário.	`string`

Chamada

```
POST /oauth_api/basic_user_info HTTPS/1.1
Host: apps.olx.com.br
Content-Type: application/json;charset=UTF-8
{
  "access_token":"1/fFAGRNJru1FTz70BzhT3Zg"
}
```

Retorno

```
{
  "user_name":"José da Silva",
  "user_email":"jose.silva@sample.com"
}
```

3.2.2 autoupload

Sistema de envio de anúncios de forma automática.

Para mais informações, consultar o manual de Autoupload do olx.com.br

4. Exemplo de utilização

A seguir, um exemplo em PHP de como utilizar nosso sistema de OAuth:

oauth.php

```
<?php
	# antes de usar este script, você deve registrar sua aplicação na OLX
	/*
	*
	*	@client_id		- token fornecido pelo OLXque representa sua aplicação
	*	@client_secret - token fornecido pelo OLXque garante as credenciais da sua
aplicação
	*
	*/
	# utilizar seu client_id e client_secret fornecido pelo OLX
	$client_id		= '52e213308e9d882584498ed90074bba58250c54e';
	$client_secret = 'db016607c4395f05df1eeb1f18e93ae2';

	$response_type = 'code';
	$scope		= 'basic_user_info';
	$redirect_uri  = 'http://127.0.0.1/oauth.php';
	$state		= 'bla';
	$grant_type		= 'authorization_code';

	$auth_host		= 'dev04c6.srv.office:22406';
	$apps_host		= 'dev04c6.srv.office:22406';
	$url		=
"https://{$auth_host}/oauth?client_id={$client_id}&response_type={$response_type}&scope={$sc
ope}&redirect_uri={$redirect_uri}&state={$state}";
?>

<!-- Requisição do código de autenticação - Como explicado na seção 2.1.1 do Manual do OAuth
-->
<!-- O usuário deve clicar no link para iniciar o 'oauth login' -->
<a href="<?php echo $url; ?>">Fazer login no olx.com.br</a><br><br>
<!-- uma vez clicado, o usuário deverá se autenticar no OLXe autorizar a sua aplicação a
acessar seus dados -->

<?php
	# isso só deve ocorrer após o recebimento do 'code' no redirect do oauth
	if (isset($_GET['code'])) {
		$code = $_GET['code'];

		# dados necessários para a requisição do token de acesso
```

```
                $fields = array(
                        'code'                => $code,
                        'client_id'           => $client_id,
                        'client_secret'       => $client_secret,
                        'redirect_uri' => $redirect_uri,
                        'grant_type'   => $grant_type
                );

                # Requisição da chave de acesso - Como explicado na seção 2.2.1 do Manual do
OAuth
                # para utilizar o método 'http_post_fields', instalar o pacote: pecl_http
                $response      = http_post_fields("https://{$auth_host}/oauth/token",
$fields);

                # separa o header do body do pacote http
                $res           = preg_split("#\n\\s*?\n#", $response, 2);
                $data          = json_decode($res[1]);
                echo "access_token: {$data->access_token}<br />";

                # salvar access_token
                #
                # O access_token deve ser salvo para evitar
                # a necessidade de fazer novamente o handshake do OAuth
                //$generic_database->saveAccessToken($data->access_token); // comando de
exemplo

                $request_data  = array('access_token' => $data->access_token);

                # Acesso à API - Como explicado na seção 3.1 do Manual do OAuth
                # neste exemplo, estamos acessando o resource basic_user_info
                # que simplesmente retorna informações básicas do usuário
                # para utilizar o método 'http_post_data', instalar o pacote: pecl_http
                $response             =
http_post_data("https://{$apps_host}/oauth_api/basic_user_info",
json_encode($request_data));

                # separa o header do body do pacote http
                $res                  = preg_split("#\n\\s*?\n#", $response, 2);
                echo "basic_user_info: {$res[1]}";

        }
?>
```

5. Referências

- ❏ [RFC6749] The OAuth 2.0 Authorization Framework [http://tools.ietf.org/html/rfc6749]
- ❏ Using OAuto 2.0 for Web Server Applications [https://developers.google.com/accounts/docs/OAuth2WebServer?hl=pt-BR]