

## Lichess API. Аутентификация

Сайт Lichess предоставляет 2 способа аутентификации, которые могут использовать программисты в своих приложениях:

1. Использование персонального токена (Personal Access Token).
2. Авторизация с PKCE (RFC 7636: Proof Key for Code Exchange).

Оба эти способа подразумевают то, что конечные пользователи зарегистрированы на Lichess.org.

### Использование Personal Access Token

Этот способ позволяет отправлять запросы без авторизации по PKCE. Получить персональный токен можно в настройках аккаунта во вкладке API access tokens.

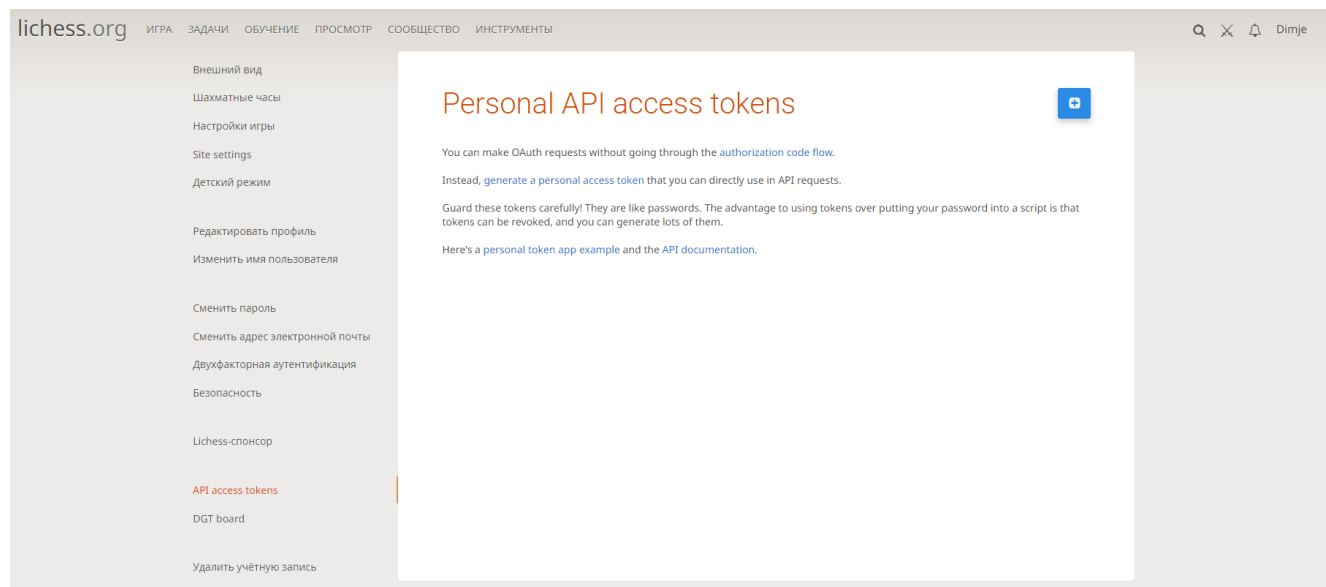


Рисунок 1. Настройки аккаунта.

После получения токена для выполнения запросов, которые требуют авторизации пользователя, необходимо использовать следующий заголовок **"Authorization: Bearer {token}"**, где token — это персональный токен.

### Авторизация с PKCE.

Получение токена в данном случае происходит при выполнении GET запроса по адресу <https://lichess.org/oauth> и передаче необходимых параметров (процесс получения токена описан в файле OAuth.docx, перевод на русский, и по адресу <https://lichess.org/api#tag/OAuth>). Токен возвращается при авторизации на Lichess. При получении токена параметр scope, указывает запрошенные области действия OAuth см. рис.2.

Security Scheme Type	OAuth2
authorizationCode OAuth Flow	<p><b>Authorization URL:</b> <a href="https://lichess.org/oauth">https://lichess.org/oauth</a>  <b>Token URL:</b> <a href="https://lichess.org/api/token">https://lichess.org/api/token</a>  <b>Scopes:</b></p> <ul style="list-style-type: none"> <li><code>preference:read</code> - Read your preferences</li> <li><code>preference:write</code> - Write your preferences</li> <li><code>email:read</code> - Read your email address</li> <li><code>challenge:read</code> - Read incoming challenges</li> <li><code>challenge:write</code> - Create, accept, decline challenges</li> <li><code>challenge:bulk</code> - Create, delete, query bulk pairings</li> <li><code>study:read</code> - Read private studies and broadcasts</li> <li><code>study:write</code> - Create, update, delete studies and broadcasts</li> <li><code>tournament:write</code> - Create tournaments</li> <li><code>puzzle:read</code> - Read puzzle activity</li> <li><code>team:read</code> - Read private team information</li> <li><code>team:write</code> - Join, leave, and manage teams</li> <li><code>follow:write</code> - Follow and unfollow other players</li> <li><code>msg:write</code> - Send private messages to other players</li> <li><code>board:play</code> - Play with the Board API</li> <li><code>bot:play</code> - Play with the Bot API. Only for <a href="#">Bot accounts</a></li> <li><code>web:mod</code> - Use moderator tools (within the bounds of your permissions)</li> </ul>

Рисунок 2. Области действия OAuth, которые можно получить.

После получения токена OAuth, его необходимо использовать так же, как и персональный токен.

## Войти

Логин или электронная почта

Dimje

Пароль

\*\*\*\*\*

ВОЙТИ

[Регистрация](#)

[Сброс пароля](#)

[Log in by email](#)



Authorize

<http://127.0.0.1>

Grant access to your **Dimje** account:

- Read preferences
- Read email address

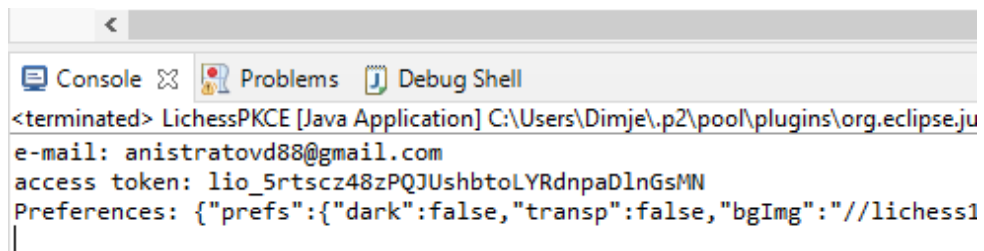
[Cancel](#)

[AUTHORIZE](#)

Not owned or operated by lichess.org

Will redirect to <http://127.0.0.1:60079>

Рисунок 3. Авторизация при помощи Lichess.



```
<terminated> LichessPKCE [Java Application] C:\Users\Dimje\.p2\pool\plugins\org.eclipse.ju
e-mail: anistratovd88@gmail.com
access token: lio_5rtszcz48zPQJUshbtoLYRdnpaDlnGsMN
Preferences: {"prefs":{"dark":false,"transp":false,"bgImg":"//lichess1
|
```

Рисунок 4. Получение email адреса аккаунта и настроек внешнего вида

Пример получения токена на Java посмотрели на <https://github.com/tors42/lichess-oauth-pkce-app>. Пример запроса на Java для получения настроек внешнего вида:

```
static String readPreferences(String access_token) throws URISyntaxException,
    IOException, InterruptedException {

    URI uri = new URI(LichessUri + "/api/account/preferences");
    var request = HttpRequest.newBuilder(uri)
        .GET()
        .header("authorization", "Bearer " + access_token)
        .header("accept", "application/json")
        .build();

    var response = HttpClient.newHttpClient().send(request,
        BodyHandlers.ofString());
    var statusCode = response.statusCode();
    var preferences = response.body();

    if (statusCode != 200) {
        System.out.println("/api/account/preferences - " + statusCode);
    }

    return preferences;
}
```

В документации также указано, какие области действия токена нужны для выполнения того или иного запроса.

AUTHORIZATIONS: OAuth2 ( challenge:read, bot:play, board:play )

Рисунок 5. Области действия токена, необходимые для выполнения запроса.

Также на сайте документации указывается, какой формат будут содержать данные ответа.

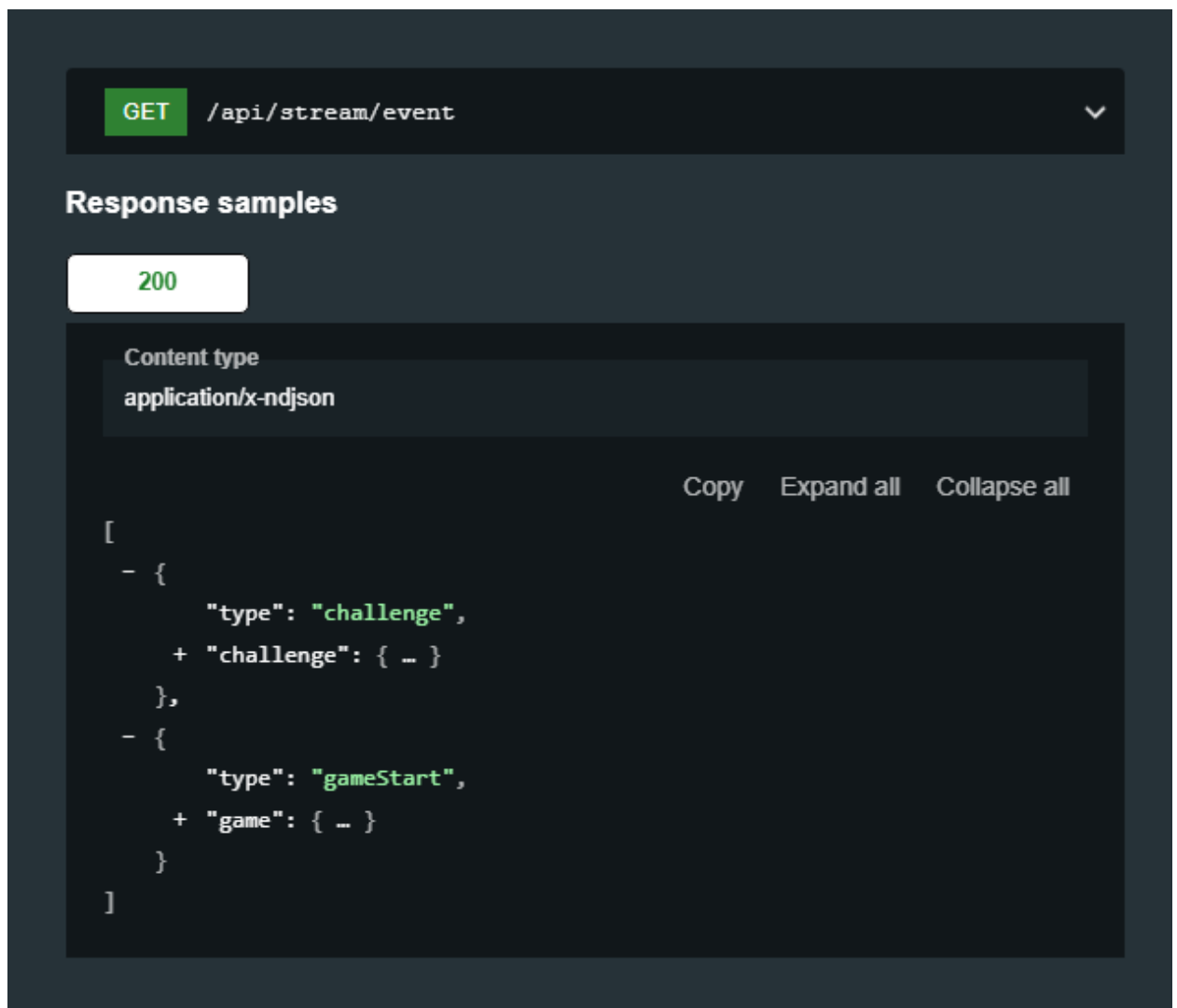


Рисунок 6. Информация о возвращаемых данных.

Кроме того в API есть адреса, запросы на которые не требуют авторизации пользователя.