ПАО Московская биржа

Подключение API

Описание интерфейса API

Версия 1.3
4.10.2023

Оглавление

Подключение API. Пошаговая инструкция

Для использования API Московской Биржи вам необходимо выполнить следующие шаги:

1. Создайте приложение для доступа к выбранным API Биржи.
2. Реализуйте в своем приложении поддержку протокола OAuth 2.0
3. Получите токен доступа (access_token – см. ниже), который вы сможете использовать для вызова функций API.
4. Протестируйте ваше приложение (см. ниже), обратившись в службу поддержки help@moex.com
5. Запросите своего персонального менеджера о присвоении приложению параметров для использования в промышленной среде *client_id* и *client_secret* (уникальный идентификатор вашего приложения и ключ безопасности, в совокупности однозначно идентифицирующие ваше приложение при обращении к API), отправив ему заявление на подключение к выбранному сервису API Биржи.
6. Необходимо учитывать, что у пользователя, указанного в заявлении на подключение, от имени которого планируется работать с API, Удостоверяющим центром Московской биржи должен быть выпущен сертификат электронного ключа на имя этого пользователя (владелец сертификата). Область действия - Московская Биржа.
7. После проверки и утверждения созданного вами приложения, а также прав пользователя, персональный менеджер направит учетные данные client_id и client_secret для доступа к запрошенным API Биржи.
8. Вы готовы использовать API Московской Биржи.

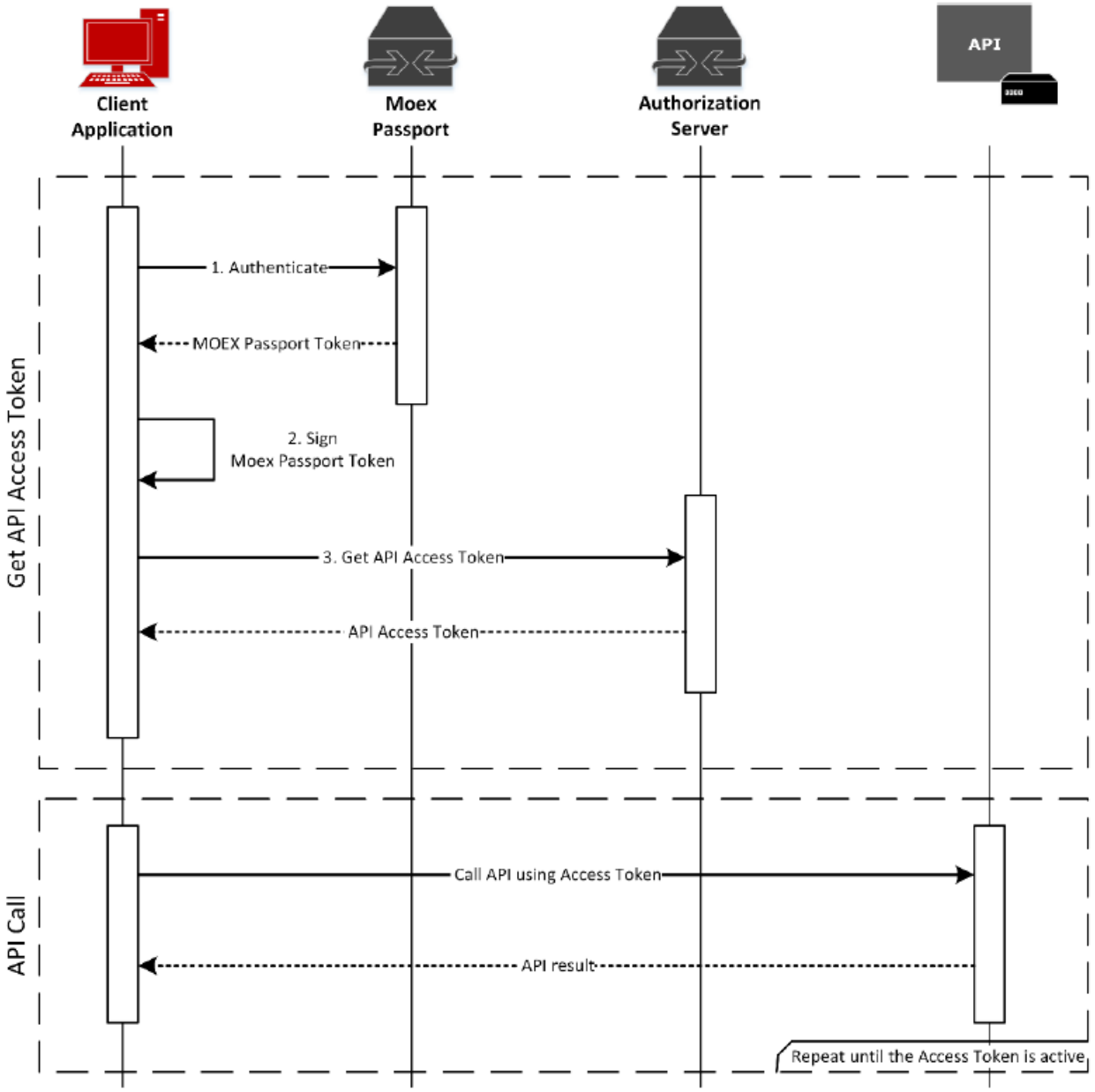
Что такое OAuth 2.0?

OAuth - открытый стандарт аутентификации и авторизации. OAuth предоставляет метод доступа клиентов к ресурсам сервера от имени владельца ресурса (такого как другой клиент или конечный пользователь). Он также обеспечивает процесс для конечных пользователей авторизации доступа третьих сторон к их ресурсам сервера без совместного использования их учетных данных.

Системные требования:

1. При использовании СКЗИ Валидата:
   - Операционная система Windows 7 и новее.
   - В настройках безопасности браузера необходимо отключить протокол tls ver 1.0 и выставить поддержку протокола tls ver 1.1 и/или tls ver 1.2 .
2. Должна устанавливаться telnet-сессия на адреса:
   https://passport.moex.com и https://apim.moex.com на порт 443/tcp;
   Для тестовой среды https://passport-test.moex.com и https://play-apim.beta.moex.com на порт 443/tcp.
3. HTTPS запросы должны быть разрешены в вашей локальной сети.
4. Не рекомендуется использование прокси-сервера.

Общая схема работы с API



Получение токена доступа

Чтобы получить токен доступа, необходимо выполнить несколько шагов:

1. Получить MOEX Passport Token, выполнив GET запрос по адресу https://passport.moex.com/authenticate, используя Basic аутентификацию с учетными данными пользователя, от имени которого предполагается работа с API. Значение Moex Passport Token будет возвращено в cookie MicexPassportCert;

2. Используя API СКЗИ Валидата (для формирования подписи в форматах ГОСТ и RSA), а также иное криптографическое ПО для формирования подписи в стандарте RSA, создать отсоединенную электронную цифровую подпись полученного на предыдущем шаге MOEX Passport Token сертификатом пользователя, от имени которого предполагается работы с API.

Всю необходимую информацию по работе с СКЗИ вы можете найти на http://moex.com/s1292

3. Выполнить POST запрос по адресу https://sso.moex.com/auth/realms/SSO/protocol/openid-connect/token, использовав следующие параметры (параметры должны передаваться с использованием метода “application/x-www-form-urlencoded”):

   - **grant_type** – password
   - **grant_type_moex** – passport
   - **scope** – запрашиваемые права доступа
   - **client_id** – идентификатор приложения, выданный вашим персональным менеджером
   - **client_secret** – ключ безопасности, выданный вашим персональным менеджером
   - **certificate** – MOEX Passport Token, полученный на первом шаге
   - **algorithm** – значение GOST или RSA, в зависимости от типа подписи, использованной при формировании электронной подписи MOEX Passport Token
   - **signature** – электронная подпись MOEX Passport Token, сформированная на втором этапе, в Base64 кодировке

Если запрос выполнится успешно, вы получите JSON объект со следующими полями:

- **access_token** – токен доступа, который должен передаваться при каждом вызове API
- **expires_in** – время жизни токена доступа в секундах
- **refresh_expires_in** – время жизни токена обновления в секундах
- **refresh_token** – токен обновления, токен который необходимо использовать при обновлении текущего токена доступа
- **token_type** – всегда имеет значение Bearer
- **not-before-policy** – активна ли политика неиспользования токена ранее установленного времени после выпуска (‘0’ – не активна)
- **session_state** – идентификатор аутентифицированной сессии
- **scope** – полученные права доступа

В случае же, если переданные данные не являются валидными (например, приложение с таким client_id отсутствует, client_secret не соответствует client_id или же переданная электронная подпись не соответствует переданному MOEX Passport токену) результатом буден **HTTP Response Code 403**

Использование токена доступа

Теперь, когда у вас есть токен доступа, все, что вам необходимо сделать, это использовать его для подписания запросов, отправленных в API, добавляя следующий заголовок к запросам:

Authorization: Bearer <access_token>

В случае, если используемый Access Token (токен доступа) не является валидным или время его жизни истекло, в ответ вы получите **HTTP Response Code 401**. При получении ответа с данным кодом ошибки, вы можете повторно запросить токен согласно инструкции, описанной выше.

Тестирование

Перед тем как осуществить первый запрос к сервису в промышленной среде, рекомендуется провести тестирование работы готового приложения. Для этого необходимо выполнить последовательность шагов:

1. Создать тестовую учетную запись на https://passport-test.moex.com/registration.
2. Получить MOEX Passport Token, выполнив GET запрос по адресу https://passport-test.moex.com/authenticate, используя Basic аутентификацию с учетными данными пользователя, от имени которого предполагается работа с API. Значение Moex Passport Token будет возвращено в cookie MicexPassportCert;
3. Прислать на адрес help@moex.com письмо с заголовком «Тестирование сервиса WEBApi *<желаемый сервис>*», в котором указать:
   - Email, который использовался для регистрации в MOEX Passport в п. 1;
   - Идентификатор фирмы, с которой вы работаете на тестовых стендах INET_GATEWAY/INETCUR_GATEWAY (если есть). Если такой фирмы нет, то она будет создана.
   - Желаемый тип сертификата для п. 4 (RSA или ГОСТ)

   В ответном письме будут высланы:
   - Тестовый сертификат пользователя для п. 4;
   - client_id и client_secret для п. 6;
   - Данные для подключения к тестовым стендам INET_GATEWAY и INETCUR_GATEWAY, если необходимы.

4. Создать откреплennую ЭЦП.

Для ГОСТ-сертификата и СКЗИ "Валидата CSP" версии 6.0: скачать дистрибутивы можно тут https://www.moex.com/s1292

- Скачать и установить Дистрибутив АПК Клиент МБ (ПК "Справочник сертификатов", ZCS) соответствующей разрядности (если не установлено/без поддержки компонент TLS, при необходимости добавить считыватель типа «Реестр»). Перезагрузить компьютер.
- Скачать и установить Дистрибутив СКЗИ "Валидата CSP" соответствующей разрядности (если не установлено). Запустить.
- Запустить Справочник сертификатов. При первичном запуске необходимо выполнить восстановление из резервной копии. В полученном в п. 2 архиве с тестовым сертификатом в папке Spr находится база локального справочника сертификатов. Эту папку нужно указать в качестве источника резервной копии. Также можно скопировать ее содержимое в каталог профиля C:\Users\<USER>\AppData\Roaming\Validata\zcs\. При следующем запуске указать тестовый сертификат. В дальнейшем при запуске Справочника будет выводиться сообщение с просьбой предоставить носитель с ключом, для этого необходимо выгрузить содержимое папки vdkeys из архива в корневой каталог съемного носителя (виртуальная дискета или USB-флэш накопитель).
- Для ГОСТ-сертификата необходимо использовать Утилиту командной строки zpki1utl. Утилита командной строки поставляется с ПК "Справочник сертификатов" и располагается по умолчанию в C:\Program Files\Validata\zpki. Пример команды для создания отсоединенной ЭП:

  zpki1utl.exe -profile <User> -sign -detached -data <token> -out token.p7d

Для **RSA**-сертификата: скачать дистрибутивы можно тут https://www.moex.com/s1293

- Скачать и установить Дистрибутив ПКЗИ СЭД МБ (ПК "Справочник сертификатов", RCS) версии не ниже 6.0.
- Запустить Справочник сертификатов. При первичном запуске необходимо выполнить восстановление из резервной копии. В полученном в п. 2 архиве с тестовым сертификатом в папке Spr находится база локального справочника сертификатов. Эту папку нужно указать в качестве источника резервной копии. Также можно скопировать ее содержимое в каталог профиля C:\Users\<USER>\AppData\Roaming\Validata\rcs\. При следующем запуске указать ключ UserOrg.rsa.
- Скачать утилиту Утилита командной строки для использования в СЭД (прямая ссылка http://fs.moex.com/cdp/po/rpkiutlv6.zip). Для RSA необходимо использовать утилиту rpki1utl* соответствующей разрядности. Краткое описание параметров запуска находится в архиве, файлы RunUtil.txt и rpki1utl.txt. Создать отделенную ЭЦП, пример команды:

  rpki1utl.exe -profile <User> -sign -detached -data <token> -out token.p7d

5. Далее полученный токен необходимо закодировать в Base64 с помощью любого инструмента или функции используемого языка программирования. Пример использования стандартной команды в Windows:

   certutil -encode token.p7d token.sig

Полученный файл подписи не должен содержать переносов строк! Соответственно, для команд, не имеющих параметра для создания файла без переносов (включая приведённый пример с certutils), необходимо их удалить из закодированного в Base64 файла.

6. Выполнить POST запрос по адресу https://sso2.beta.moex.com/auth/realms/SSO/protocol/openid-connect/token, использовав следующие параметры (параметры должны передаваться с использованием метода “application/x-www-form-urlencoded”):
   - **grant_type** – password
   - **grant_type_moex** – passport
   - **scope** – запрашиваемые права доступа
   - **client_id** – новый идентификатор приложения
   - **client_secret** – новый ключ безопасности
   - **certificate** - MOEX Passport Token, полученный как в п. 2
   - **algorithm** – значение GOST или RSA, в зависимости от типа подписи, использованной при формировании электронной подписи MOEX Passport Token
   - **signature** – электронная подпись MOEX Passport Token в Base64 кодировке, сформированная как в п. 4

Общий алгоритм двухфакторной аутентификации:

Общий алгоритм двухфакторной аутентификации:

MicexPassportCert = Аутентифицироваться_MOEX_Passport (login, password) ;

Signature = Валидата.создать_ЭЦП (MicexPassportCert , CLIENT_CERTIFICATE);

access_token = HTTP.post (url=https://sso2.beta.moex.com/auth/realms/SSO/protocol/openid-connect/token, параметры= {

grant_type= “password”, grant_type_moex= “passport”, scope “*<scope>*”,

client_id=*Client_id*, client_secret=*Client_secret*,

certificate= MicexPassportCert,

algorithm=”GOST”|”RSA”, signature=Signature

}

Если запрос выполнится успешно, вы получите JSON объект со следующими полями:

- **access_token** – токен доступа, который должен передаваться при каждом вызове API
- **expires_in** – время жизни токена доступа в секундах
- **refresh_expires_in** – время жизни токена обновления в секундах
- **refresh_token** – токен обновления, токен который необходимо использовать при обновлении текущего токена доступа
- **token_type** – всегда имеет значение Bearer
- **not-before-policy** – активна ли политика неиспользования токена ранее установленного времени после выпуска (‘0’ – не активна)
- **session_state** – идентификатор аутентифицированной сессии
- **scope** – полученные права доступа

В случае же, если переданные данные не являются валидными (например, приложение с таким client_id отсутствует, client_secret не соответствует client_id или же переданная электронная подпись не соответствует переданному MOEX Passport токену) результатом буден **HTTP Response Code 403**.