

# TalentTech API

---

- [Глоссарий](#)
- [Общая схема данных](#)
- [Получение доступа к API](#)
  - [Регистрация приложения](#)
    - [Скоупы приложения](#)
  - [Получение токена](#)
- [WebHook](#)
  - [Авторизация хука](#)
  - [Параметры хука](#)
- [OpenAPI спецификации](#)
  - [Core API](#)

## Глоссарий

---

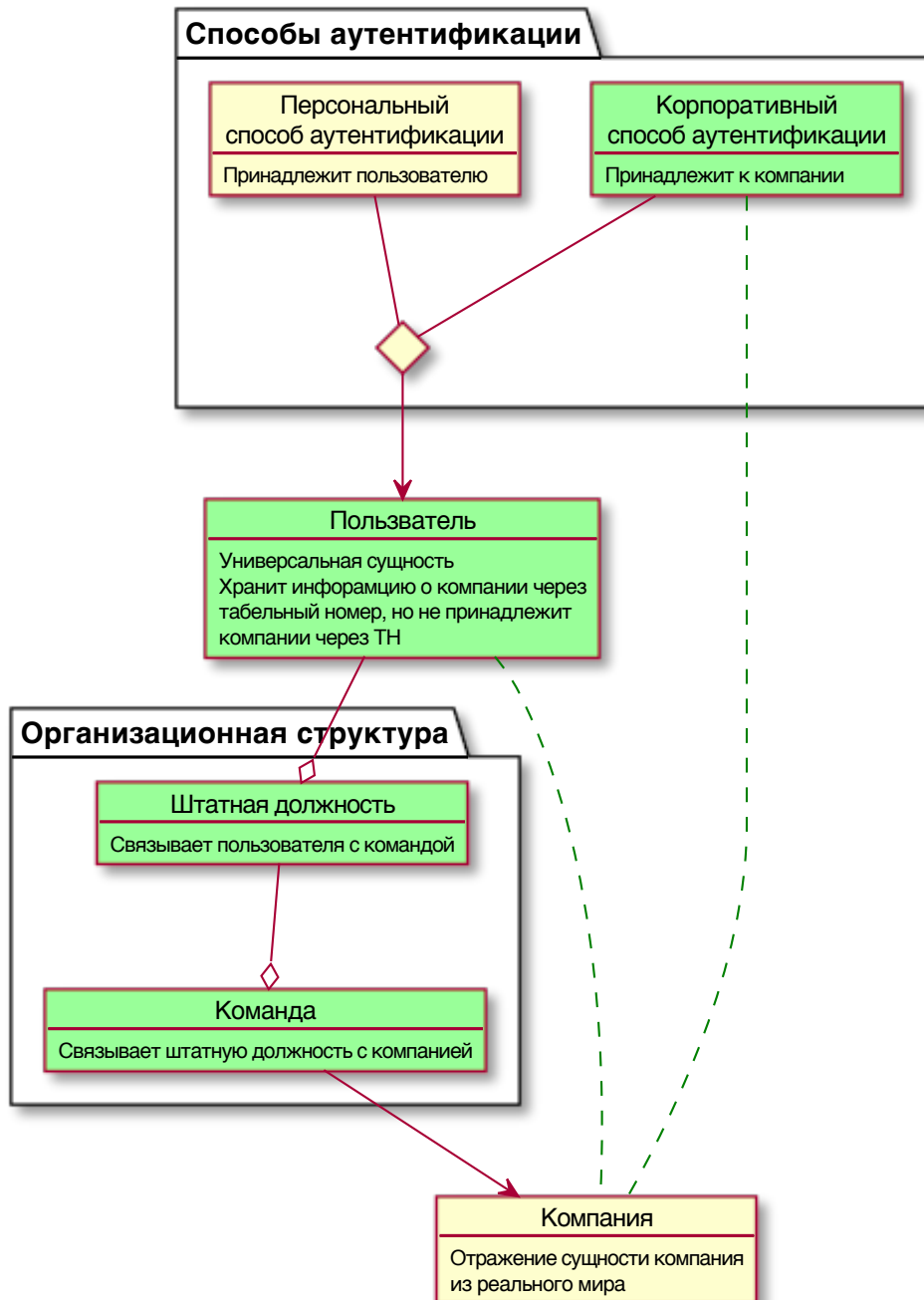
- **API платформы** - api которое предоставляет платформа вне зависимости от используемых модулей. Данное API необходимо использовать для работы с базовыми сущностями платформы, такими как: пользователь, команда и т.д.
- **API модуля** - api которое предоставляет непосредственно модуль, данной api варьируется от модуля к модулю (некоторые модули не имеют api)
- **Приложение** - oAuth приложение которое необходимо для осуществления внешних интеграций с платформой ТТ. С приложением ассоциирован ключ и набор скоупов которые позволяют выполнять различные наборы вызовов. Также приложения можно использовать в качестве OpenID Identity Provider для OpenID Connect интеграций.
- **Ключ** - При создании приложения генерируется пара ключей для асимметричного шифрования (открытый и закрытый ключи). Открытый ключ хранится на платформе ТТ и используется для проверки запросов которые отправляются в API и подписываются закрытым ключом который хранится на стороне разработчика приложения. Закрытый ключ показывается только 1 раз и нигде не сохраняется, его можно только заново создать. Хранить его обязанность стороны разработчика приложения.
- **Токен** - Токен который получен с использованием закрытого ключа и `auth` API. Для осуществления API вызовов необходим этот токен. Данный токен имеет срок действия и его требуется получать заново после окончания срока действия. Более подробно этот процесс рассмотрен в разделе [получение токена](#).
- **Компания** - Платформа ТТ и модули - это облачное решение. Для правильного разделения данных в рамках облака все клиенты платформы действуют в рамках компании. Каждая компания (обычно, но не обязательно) является отражением компании в реальном мире.

- **Пользователь** - Пользователи на платформе ТТ не имеют непосредственной связи с компанией и могут находиться в нескольких компаниях сразу. Для корректного разделения доступа к данным пользователя существует два механизма
  - **Табельный номер** - если пользователь имел какое-либо отношение к компании то он может иметь табельный номер в данной компании. Это позволяет даже после увольнения пользователя из компании, работать с его данными.
  - **Организационная структура** - это единственный способ связать активного пользователя с компанией. Помещая пользователей на штатные должности в компании вы фактически осуществляете факт приёма сотрудников на работу и таким образом они связываются с компанией.
- **Способ аутентификации** - Платформа ТТ для аутентификации пользователей имеет множество механизмов и эти механизмы постоянно добавляются. Одним из наиболее простых являются одноразовые токены через посылку sms сообщений на телефон или кодов в email сообщениях. Таким образом пользователь не должен помнить пароль, а работодателю легко внести учетные записи пользователя в систему без организации сложных рассылок данных для входа. Способы аутентификации тоже делятся на 2 категории, что обусловлено облачной природой платформы ТТ.
  - **Персональные** - способы аутентификации которые принадлежат лично пользователю, т.е. они были добавлены самим пользователем и управляются им самим.
  - **Корпоративные** - те которые были добавлены компанией и управляются компанией. Пользователь не может ими управлять в отличие от персональных.
- **Организационная единица (команда)** - структура любой компании состоит из отделов и подразделов. Организационная единица на платформе ТТ является отражением этой сущности. Платформа позволяет создать сколько угодно корневых команд и организовать какую угодно вложенность команд.
- **Штатная должность** - для помещения человека в команду существует штатная должность, таким образом выстраивается связь между пользователем платформы и компанией в которой он работает (если пользователь находится на штатной должности значит он находится в компании). При этом штатную должность может занимать только один пользователь (если в отделе 10 человек, значит должно быть минимум 10 штатных должностей). Штатные должности могут быть вакантными (в этом случае на них просто нет назначенного пользователя). Все связи (руководитель -> подчинённый) выстраиваются при помощи связи штатных должностей между собой, что позволяет например при смене руководителя отдела, не перестраивать связи между руководителем и подчинёнными.
- **Организационная структура компании** - совокупность команд и штатных должностей в компании.

## Общая схема данных платформы TalentTech

Схема ниже показывает отношения между сущностями и организацию их по группам. Из схемы видно как способы аутентификации связаны с пользователем и каким образом пользователь непосредственно связан с компанией.

Объекты зеленого цвета - это те к которым у разработчиков приложений есть доступ через API платформы. Другие же объекты недоступны для управления.



## Получение доступа к API

Для получения доступа к API платформы ТТ необходимо совершить два действия:

- Зарегистрировать приложение
- Получить JWT токен для осуществления вызовов

Для авторизации действий приложений используется OAuth2 и концепция сервисных аккаунтов. Это означает, что каждое приложение действует не от лица определённого пользователя, а от лица компании. При этом возможность совершать действия от лица определённого пользователя предусмотрена и может быть реализована на уровне API модуля (это зависит исключительно от модуля).

Для возможности выполнения действия от имени пользователя у приложения должен быть скоуп: `user:action`, а при отправке запроса необходимо указать идентификатор пользователя в заголовке `x-User-ID`.

Для авторизации API вызовов используется стандартный заголовок `Authorization` с указанием типа токена. Мы всегда используем `Bearer` так что наш заголовок выглядит всегда очень одинаково:

```
Authorization: Bearer <token>
```

Для получения авторизационного токена необходимо выполнить следующие действия:

- Зарегистрировать приложение и получить приватный ключ для подписи запроса на получение токена
- Отправить запрос на получение токена через API и подписать этот запрос приватным ключом из первого пункта

## Регистрация приложения

Для подключения приложения нужно пройти стандартную процедуру регистрации приложения, получения `client_id` и ключей.

- Зайти в интерфейс управления приложениями
- Добавить приложение - получить `client_id`
- Выбрать скоупы в которых может действовать приложение (скоупы платформы и модулей)
- Получить пару ключей для подписи и проверки подписи токенов.

После регистрации приложения нужно выполнить стандартную процедуру получения ключа:

- Сформировать JWT токен
- Запросить токен у сервера авторизации
- Использовать токен полученный на предыдущем шаге для API вызовов

Токен полученный для API вызовов имеет определённый срок действия, если срок действия подошёл к концу, то процедуру получения токена необходимо произвести снова.

## Скоупы приложения

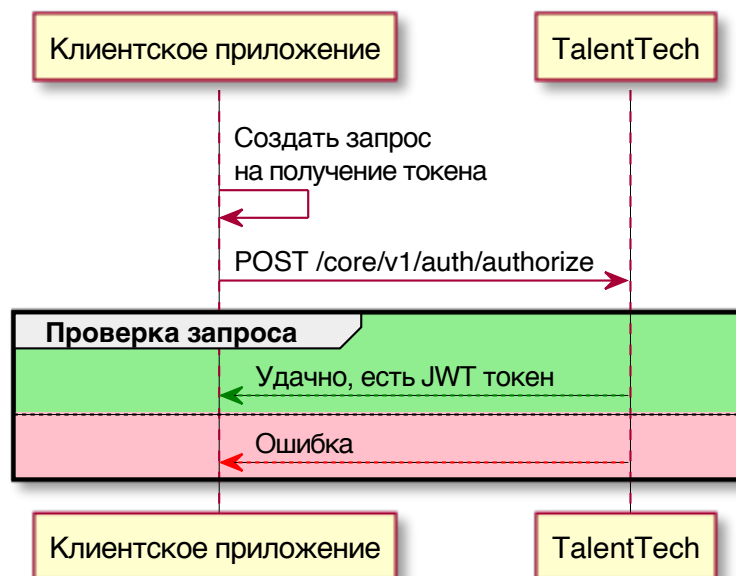
Платформа и модули реализуют определённый набор скоупов в рамках которых приложения могут выполнять определённые действия. Возможность регистрировать много приложений, а также возможность разделять их на скоупы позволяет построить на стороне клиента максимально гибкую систему в плане безопасности.

- `users` - доступ к данным пользователей
  - `users:read` - чтение данных пользователей
  - `user:write` - запись данных пользователей
  - `user:action` - возможность выполнять действия от имени пользователя в системе
- `auth:write` - специальный скоуп для управления аутентификационными данными пользователей, читать их можно в скоупе `users:read`
- `teams` - доступ к организационной структуре компании

- teams:read - чтение данных организационной структуры
- teams:write - запись данных организационной структуры

## Получение токена

После регистрации приложения и получения ключа, необходимо получить токен который используется непосредственно для выполнения API запросов (тот самый который передаётся в заголовке `Authorization`). Это обычный токен в формате JWT который содержит информацию об окончании срока действия. После того как токен перестаёт быть валидным клиент обязан повторить процедуру получения токена. Процедура получения выглядит следующим образом:



## Формирование запроса на получение токена

При формировании запроса на получение токена, клиент должен сформировать JWT токен со следующим содержанием:

```
{
  "iss" : "5b6b2470-d1b0-0139-a07d-7be4b2c2fab5",
  "exp" : 1627462923,
  "alg" : "RS256"
}
```

iss	Уникальный идентификатор приложения которое запрашивает токен - client_id
exp	Дата окончания срока действия токена, должна быть не более чем 30 секунд от момента формирования токена
alg	Алгоритм подписи токена, т.к. используется асимметричное шифрование и клиент подписывает токен своим ключом, необходимо использовать: SHA-256 (RS256)

При формировании токена, клиент должен подписать его своим приватным ключом который он получил при регистрации приложения. Токен следует передавать в параметре `token` в `POST` параметрах запроса.

## Токен для авторизации API вызовов

В ответ клиент получит новый JWT токен который он должен использовать без каких-либо модификаций в заголовке `Authorization` для авторизации API вызовов. Токен будет содержать следующие поля:

```
{
  "iss" : "9478afdf-3116-45e0-86f3-0a0c69c8cdd7",
  "exp" : 1627462923,
  "alg" : "HS256"
}
```

iss	Идентификатор токена.
exp	Дата окончания срока действия токена, после неё токен недоступен для использования и нужно повторить процедуру получения токена заново.
alg	Алгоритм подписи токена.

## WebHook

Для обеспечения возможности обратной связи между сервисами TalentTech и внешними интеграциями. Предусмотрен механизм регистрации webhook'ов. Каждое приложение может зарегистрировать не ограниченное количество хуков.

Для обеспечения возможности обратной связи между платформой TalentTech и клиентскими приложениями предусмотрен механизм регистрации WebHook'ов. Клиентское приложение может зарегистрировать адрес для получения информации об асинхронных событиях происходящих в платформе. Адрес всегда один и в него попадает вся информация. В качестве формата передачи используется `json`.

## Авторизация хука

При регистрации хука следует указать секретный токен на основании которого будет вычисляться подпись и передаваться в заголовке `Authorization`, что позволит принимающей стороне однозначно проверить подлинность вызова.

В заголовке передаётся SHA256 хеш от секретного токена и клиентская сторона обязана на своей стороне проверить, что секрет совпадает.

## Параметры хука

Хук всегда отправляет POST запрос на указанный адрес. Внутри POST запрос содержит тело в виде JSON сообщения.

Сообщение содержит следующие ключи

Ключ	Описание
<code>company_id</code>	Уникальный идентификатор компании в рамках которой произошло событие.
<code>user_id</code> *	Уникальный идентификатор пользователя от которого было выполнено действие. Этот идентификатор присутствует в сообщении только в том случае, если при совершении запроса использовался заголовок <code>X-User-ID</code> и значение этого заголовка было применимо в запросе.
<code>status</code>	Статус выполнения операции, может принимать следующие значения: <code>success</code> <code>error</code> <code>info</code>
<code>module</code>	Имя модуля в котором произошло событие.
<code>event</code>	Уникальное имя события в рамках модуля.
<code>description</code>	Текстовое описание события
<code>created_at</code>	Время первичного возникновения события в системе. Фиксируется именно в момент возникновения.
<code>scheduled_at</code>	Время первой попытки отправки события
<code>retries</code>	Количество попыток отправки события, первая отправка всегда содержит <code>0</code>

При регистрации хука можно указать фильтр по полям `module` и `event` , и в результате получать события которые возникают только в определённом модуле(ях) с определёнными идентификаторами.

Все дополнительные ключи которые приходят в сообщении зависят от события и описываются отдельно.

## OpenAPI спецификации

Все спецификации в формате OpenAPI 3.0.1, для просмотра можно воспользоваться публичными инструментами на <https://swagger.io/>

## Core API

Версия	V1
Файл спецификация	<a href="#">core.yaml</a>
Базовый адрес для всех вызовов	<code>/core/v1</code>

## Авторизация

Базовый путь: `/auth/`

Методы необходимые для получения JWT токена который необходимо использовать для отправки других API вызовов.

## Пользователи

Базовый путь: `/user/`

Методы для работы непосредственно с пользователем, его метаданными и способами аутентификации.

## Организационная структура

Базовый путь: `/team/`

Методы для работы с организационной структурой компании состоящей из команд и штатных должностей. При помощи методов из этого раздела осуществляется как добавление пользователей в компанию так и исключение из неё.

## Данные цифрового профиля

Базовый путь: `/dp/`

Специальный раздел платформы ТТ для приёма данных на основании которых строится цифровой профиль сотрудника. Хотя это и публичное API, но для его использования необходимо пройти процедуру описания данных т.к. без этой процедуры платформа не сможет понять какие данные вы пытаетесь отправить.