

# **Anabix API**

Popis způsobu používání služby

# Obsah

1	Základní informace.....	3
1.1	Požadavky na klientský systém.....	3
1.2	Zasílání requestů.....	3
1.3	Získávání odpovědí ze systému.....	3
1.4	Odpověď po neúspěšném zpracování.....	3
1.5	Standardní operace.....	3
1.6	Důležitá upozornění.....	5
1.7	Vzorový příklad připojení z PHP.....	5
2	Přístup ke komponentám systému.....	8
2.1	Kontakty.....	8
2.1.1	Struktura datové části requestu pro vložení nebo aktualizaci záznamu.....	8
2.1.2	Struktura vrácených informací o záznamu.....	9
2.1.3	Změna seznamů, do kterých je kontakt zařazen.....	10
2.2	Firmy.....	11
2.2.1	Struktura datové části requestu pro vložení nebo aktualizaci záznamu.....	11
2.2.2	Struktura vrácených informací o záznamu.....	11
2.3	Seznamy.....	13
2.3.1	Struktura datové části requestu pro vložení nebo aktualizaci záznamu.....	13
2.3.2	Struktura vrácených informací o záznamu.....	13
2.4	Aktivity.....	14
2.4.1	Struktura datové části requestu pro vložení nebo aktualizaci záznamu.....	14
2.4.2	Struktura vrácených informací o záznamu.....	14
2.5	Obchodní případy.....	16
2.5.1	Struktura datové části requestu pro vložení nebo aktualizaci záznamu.....	16
2.5.2	Struktura vrácených informací o záznamu.....	16
2.5.3	Šablony obchodních případů.....	17
2.6	Úkoly.....	18
2.6.1	Struktura datové části requestu pro vložení nebo aktualizaci záznamu.....	18
2.6.2	Struktura vrácených informací o záznamu.....	18
2.6.3	Šablony úkolů.....	19
2.7	Uživatelé.....	20
2.7.1	Získání seznamu aktivních uživatelů.....	20
2.8	Vlastní pole.....	21
2.8.1	Získání seznamu vlastních polí.....	21
2.9	Webhooky.....	22
3	Často kladené otázky (FAQ).....	23

## 1 Základní informace

Anabix API je služba umožňující uživatelům pomocí zasílaných požadavků vzdáleně spouštět funkce systému Anabix. Systém standardně pracuje s kódováním UTF-8.

### 1.1 Požadavky na klientský systém

Použitý programovací jazyk musí mít podporu pro přenos dat s podporou protokolu HTTP. Nejznámější knihovnou tohoto druhu je pravděpodobně cURL. Alternativou na platformě .NET je například třída `HttpWebRequest`. Dále je vhodná podpora datového formátu JSON, který je použit pro veškerou komunikaci.

### 1.2 Zasílání requestů

Veškerá komunikace je realizovaná pomocí HTTP protokolu. Žádosti jsou odesílány na předem určenou adresu jako POST požadavek obsahující JSON data s potřebnými údaji (může být volitelně uvnitř parametru `json`).

Uživatelské jméno a API token si můžete editovat na stránce Nástroje - Hlavní nastavení (<https://app.anabix.cz/anabix/settings/index>).

URL API je <https://app.anabix.cz/api>.

V příkladech v tomto dokumentu budou jako autentizační údaje použity **fiktivní** přístupy:

- Username: `example`
- Usertoken: `c0d646874b2153d236a4d1816e3501176c69c5ce`

### 1.3 Získávání odpovědí ze systému

Odpověď systému se nachází v návratové hodnotě funkce odesílající žádost (v případě PHP se jedná o funkci `curl_exec()`).

### 1.4 Odpověď po neúspěšném zpracování

Pokud byla zaslána neplatná žádost (důvodem mohou být např. neplatné přihlašovací údaje, chybějící povinné pole, neplatná hodnota, nepovolený znak, nevalidní data atd.), systém vrátí stavovou zprávu ve formátu JSON, která má následující formát:

- `response`
  - `status` - stav, v případě neúspěchu bude mít hodnotu `ERROR`
  - `data` - textová zpráva vysvětlující, proč se operace nezdařila

### 1.5 Standardní operace

Všechny komponenty přístupné přes API mají k dispozici základní metody:

**Vytvoření nového záznamu:** `create`

**Aktualizace záznamu:** `update`

## 1.5 Standardní operace

### Získání informací o záznamu: `get`

Při volání metody `get` lze použít volitelný parametr `fullInfo [1|0]` - v případě zadání tohoto parametru s hodnotou 1 budou o každém záznamu vráceny podrobné informace (např. pro kontakty se budou vracet i jejich aktivity, úkoly). Použití této volby může znamenat zvýšení datové náročnosti a zpomalení odpovědi serveru.

### Smazání záznamu: `delete`

### Získání více záznamů: `getAll`

Při volání metody `getAll` lze použít volitelné parametry:

- `limit` - maximální počet záznamů v odpovědi; standardní a maximální hodnota je 200
- `offset` - počet záznamu k „přeskočení“
- `criteria` - filtrování podle zadaného sloupce ve tvaru `<název sloupce> => <požadovaná hodnota>`; je možné filtrovat podle více kritérií současně.

V textových polích lze vyhledávat v obsahu použitím syntaxe `%výraz%`.

Pole s datem a čísla lze také filtrovat pomocí rozmezí `from – to`.

Např. výběr záznamů podle data vytvoření se realizuje pomocí "criteria":

```
{"createdTimestamp":{"from":1383242003,"to":1385415197}}, (v PHP jako 'criteria' => array('createdTimestamp' => array('from' => 1383242003, 'to' => 1385415197))).
```

Filtrovat lze i pomocí hodnot vlastních polí, kdy název sloupce je definován jako `customField-[ID pole]`, tj příkladem `{"customField-128" : 1}`. U vlastních polí typu *date* lze aplikovat také rozmezí `from – to`. Seznam definovaných vlastních polí a jejich ID lze získat pomocí API, viz dokumentace níže, metoda `getCustomFields`.

- `orderBy` - řazení podle zadaného sloupce ve tvaru `<název sloupce> => <způsob řazení>` (ASC nebo DESC); je možné řadit podle více kritérií současně.

Příklad: `"orderBy":{"idOrganization":"ASC"}` (v PHP `'orderBy' => array('idOrganization' => 'ASC')`)

- `fullInfo [1|0]` - v případě zadání tohoto parametru s hodnotou 1 budou o každém záznamu vráceny podrobné informace (např. pro kontakty se budou vracet i jejich aktivity, úkoly)
- `includeMetadata [1|0]` - v případě zadání s hodnotou 1 budou do odpovědi serveru vloženy informace o celkovém počtu záznamů vyhovujících zadaným kritériím a také údaje *offset* a *limit*, kterými byl výsledek redukován. Toto je vhodné například pro implementaci stránkování.

Např: `"metadata": {"totalRecords": "1261", "offset": 0, "limit": 1}`

- `idAsKey [1|0]` - Je-li nastaveno na 0, server nevrací ID záznamů jako klíč. To je vhodné např. při parsování pomocí JS. Lze použít i při volání API metody `get`. Výchozí hodnota je 1.

Všechny typy záznamu lze řadit a vyhledávat podle všech hodnot, které lze k záznamu nadefinovat (tedy u kontaktu např. `firstName`, `lastName`, `email` atd.) včetně vlastních polí.

## 1.5 Standardní operace

### Získání seznamu a struktury volitelných polí: `getCustomFields`

Metoda získá seznam a strukturu všech vlastních polí definovaných na komponentě, včetně definic položek seznamů.

Více informací viz [kapitola 2.8](#).

## 1.6 Důležitá upozornění

Všechny základní metody (`create`, `update`, `get`, `delete`) vrací informace o daném záznamu. Konkrétní struktura je uvedena vždy v podkapitole dané komponenty s názvem *Struktura vrácených informací o záznamu*.

Vrácené informace o záznamu obsahují vždy datovou část `revisionInfo` s následující strukturou:

- `createdTimestamp` - časové razítko vytvoření záznamu
- `createdIdUser` - ID autora záznamu
- `createdUsername` - uživatelské jméno autora záznamu
- `updatedTimestamp` - časové razítko poslední aktualizace
- `updatedIdUser` - ID autora poslední změny záznamu
- `updatedUsername` - uživatelské jméno autora poslední změny záznamu

Při aktualizaci, mazání nebo získání informací o záznamu je nutné v request uvést ID záznamu. Název klíče je uveden vždy na začátku kapitoly příslušné komponenty.

Povinné parametry pro vkládání a záznamu jsou uvedeny vždy na začátku podkapitoly dané komponenty. Ostatní parametry jsou nepovinné. V případě jejich neuvedení při vytváření nového záznamu budou ponechány prázdné, případně do nich bude doplněna výchozí hodnota systému. V případě jejich neuvedení při aktualizaci záznamu bude zachována původní hodnota.

## 1.7 Vzorový příklad připojení z PHP

Následující příklad demonstruje připojení k API pomocí PHP. Jedná se o sled základních operací nad jedním seznamem (vytvoření, aktualizace, získání informací, smazání).

```
<?php

define('ANABIX_API_URL', 'https://app.anabix.cz/api');
define('ANABIX_API_USERNAME', 'example');
define('ANABIX_API_TOKEN', 'c0d646874b2153d236a4d1816e3501176c69c5ce');

// INICIALIZACE CURL
$ch = curl_init(ANABIX_API_URL);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_POST, 1);

// VYTVORENI NOVEHO SEZNAMU
$requestData = array(
    'username' => ANABIX_API_USERNAME,
    'token' => ANABIX_API_TOKEN,
    'requestType' => 'lists',
```

## 1.7 Vzorový příklad připojení z PHP

```
'requestMethod' => 'create',
'data' => array(
    'title' => 'Testovací seznam'
)
);
curl_setopt($ch, CURLOPT_POSTFIELDS, array('json' => json_encode($requestData)));
$response = curl_exec($ch);
$responseData = json_decode($response, true);
var_dump($responseData);

$idList = $responseData['data']['idList'];
// AKTUALIZACE SEZNAMU
$requestData = array(
    'username' => ANABIX_API_USERNAME,
    'token' => ANABIX_API_TOKEN,
    'requestType' => 'lists',
    'requestMethod' => 'update',
    'data' => array(
        'idList' => $idList,
        'title' => 'Testovací seznam - nový název'
    )
);
curl_setopt($ch, CURLOPT_POSTFIELDS, array('json' => json_encode($requestData)));
$response = curl_exec($ch);
$responseData = json_decode($response, true);
var_dump($responseData);

// ZISAKNI INFORMACI O SEZNAMU
$requestData = array(
    'username' => ANABIX_API_USERNAME,
    'token' => ANABIX_API_TOKEN,
    'requestType' => 'lists',
    'requestMethod' => 'get',
    'data' => array(
        'idList' => $idList,
    )
);
curl_setopt($ch, CURLOPT_POSTFIELDS, array('json' => json_encode($requestData)));
$response = curl_exec($ch);
$responseData = json_decode($response, true);
var_dump($responseData);

// VYPIS VSECH EXISTUJICICH SEZNAMU S POZADOVANÝMI PARAMETRY
$requestData = array(
    'username' => ANABIX_API_USERNAME,
    'token' => ANABIX_API_TOKEN,
    'requestType' => 'lists',
    'requestMethod' => 'getAll',
    'data' => array(
        'criteria' => array(
            'title' => 'Testovací seznam - nový název',
            'important' => 0
        ),
        'limit' => 10,
        'offset' => 0
    )
);
curl_setopt($ch, CURLOPT_POSTFIELDS, array('json' => json_encode($requestData)));
$response = curl_exec($ch);
```

## 1.7 Vzorový příklad připojení z PHP

```
$responseData = json_decode($response, true);
var_dump($responseData);

// SMAZANI SEZNAMU
$requestData = array(
    'username' => ANABIX_API_USERNAME,
    'token' => ANABIX_API_TOKEN,
    'requestType' => 'lists',
    'requestMethod' => 'delete',
    'data' => array(
        'idList' => $idList,
    )
);
curl_setopt($ch, CURLOPT_POSTFIELDS, array('json' => json_encode($requestData)));
$response = curl_exec($ch);
$responseData = json_decode($response, true);
var_dump($responseData);
```

## 2 Přístup ke komponentám systému

### 2.1 Kontakty

**Název komponenty mající na starosti příslušnou oblast:** `contacts`

**Název identifikátoru pro účely aktualizace nebo mazání:** `idContact`

**Povinné parametry pro uložení nového záznamu:** alepoň jedna hodnota z trojice `firstName`, `lastName`, `email`

#### 2.1.1 Struktura datové části requestu pro vložení nebo aktualizaci záznamu

- `firstName` - křestní jméno
- `lastName` - příjmení
- `email`, `email2`, `email3` – primární email a další emaily
- `phoneNumber`, `phoneNumber2`, `phoneNumber3` - telefonní číslo
- `cellNumber` - mobil
- `sex` - pohlaví
- `salutation` - oslovení
- `position` - pracovní pozice
- `primaryContact` - příznak, zda je kontakt veden jako primární ([0 | 1])
- `idOwner` - ID vlastníka kontaktu
- `vip` - příznak, zda se jedná o VIP kontakt ([0 | 1])
- `source` - zdroj kontaktu
- `website`
- `description` – popis kontaktu
- `shippingStreet`, `shippingCity`, `shippingCode`, `shippingCountry` – doručovací adresa kontaktu (ulice, město, psč, země)
- `organization` – ID, název nebo email firmy, pod kterou kontakt spadá
- `customFields` - vlastní pole
  - `id => value` (ID vlastního pole => hodnota vlastního pole)
  - v případě vlastního pole typu `selectbox` lze do hodnoty vlastního pole vložit buď ID položky seznamy (doporučeno), nebo skutečnou hodnotu
- `lists` - seznamy, do kterých má kontakt patřit
  - ID nebo název seznamu 1
  - ID nebo název seznamu 2
  - ...



## 2.1 Kontakty

- `gdprReason` - GDPR - právní důvod zpracování OÚ. Může nabývat těchto hodnot:
  - 1: Bez souhlasu
  - 2: Oprávněný zájem správce
  - 3: Plnění právní povinnosti
  - 4: Plnění smlouvy nebo jednání o jejím uzavření
  - 5: Souhlas se zpracováním osobních údajů
  - 6: Veřejný zájem či výkon veřejné moci
  - 7: Životně důležitý zájem
- `gdprAcceptanceDate` - GDPR - Datum udělení souhlasu ve formátu YYYY-MM-DD

### 2.1.2 Struktura vrácených informací o záznamu

- `idContact` - ID kontaktu
- `title` - zobrazované jméno (vytvořeno z křestního jména a příjmení, případně emailu)
- `firstName` - křestní jméno
- `lastName` - příjmení
- `email`, `email2`, `email3` - emaily
- `phoneNumber`, `phoneNumber2`, `phoneNumber3` - telefonní čísla
- `cellNumber` - mobil
- `sex` - pohlaví
- `salutation` - oslovení
- `position` - pracovní pozice
- `primaryContact` - příznak, zda je kontakt veden jako primární ([0 | 1])
- `vip` - příznak, zda se jedná o VIP kontakt ([0 | 1])
- `source` - zdroj kontaktu
- `website`
- `description` - popis kontaktu
- `shippingStreet`, `shippingCity`, `shippingCode`, `shippingCountry` – doručovací adresa kontaktu (ulice, město, psč, země)
- `organization/idOrganization` - firma, pod kterou kontakt spadá
- `customFields` - informace o vlastních polích
- `lists` - seznamy, do kterých kontakt patří
- `activities` - aktivity přiřazené ke kontaktu
- `deals` - obchodní případy přiřazené ke kontaktu
- `tasks` - úkoly přiřazené ke kontaktu

## 2.1 Kontakty

- `gdpr` - informace o právním důvodu uchování kontaktu a datu potvrzení

### 2.1.3 Změna seznamů, do kterých je kontakt zařazen

Název komponenty mající na starosti příslušnou oblast: `contacts`

Název metody: `manageLists`

#### Struktura datové části

- `idContact` - ID kontaktu  
*nebo*
- `email` - emailová adresa kontaktu
- `addTo` - seznamy, do kterých má být kontakt přidán
  - ID seznamu 1
  - ID seznamu 2
  - ...
- `removeFrom` - seznamy, ze kterých má být kontakt odebrán
  - ID seznamu 1
  - ID seznamu 2
  - ...

## 2.2 Firmy

### 2.2 Firmy

**Název komponenty mající na starosti příslušnou oblast:** organizations

**Název identifikátoru pro účely aktualizace nebo mazání:** idOrganization

**Povinné parametry pro uložení nového záznamu:** title (název firmy)

#### 2.2.1 Struktura datové části requestu pro vložení nebo aktualizaci záznamu

- title - název firmy
- body - popis firmy
- accountType - typ firmy (Customer | Potential customer | Partner | Media | Other)
- idNumber - IČ nebo jiné unikátní identifikační číslo
- vatNumber - DIČ
- phoneNumber, phoneNumber2, phoneNumber3 - telefonní čísla
- email, email2, email3 – emaily
- emailDomain - emailová doména
- website - adresa webu
- billingStreet - fakturační adresa - ulice
- billingCity - fakturační adresa - město
- billingCode - fakturační adresa - PSČ
- billingCountry - fakturační adresa - stát
- shippingStreet - doručovací adresa - ulice
- shippingCity - doručovací adresa - město
- shippingCode - doručovací adresa - PSČ
- shippingCountry - doručovací adresa - stát
- idOwner - ID vlastníka firmy
- customFields - vlastní pole
  - id => value (ID vlastního pole => hodnota vlastního pole)
  - v případě vlastního pole typu selectbox lze do hodnoty vlastního pole vložit buď ID položky seznamy (doporučeno), nebo skutečnou hodnotu

#### 2.2.2 Struktura vrácených informací o záznamu

- idOrganization - ID firmy
- title - název firmy
- body - popis firmy
- accountType - typ firmy (Customer | Potential customer | Partner | Media | Other)

## 2.2 Firmy

- `idNumber` - IČ nebo jiné unikátní identifikační číslo
- `vatNumber` - DIČ
- `phoneNumber`, `phoneNumber2`, `phoneNumber3` - telefonní čísla
- `email`, `email2`, `email3` – emaily
- `emailDomain` - emailová doména
- `website` - adresa webu
- `billingStreet` - fakturační adresa - ulice
- `billingCity` - fakturační adresa - město
- `billingCode` - fakturační adresa - PSČ
- `billingCountry` - fakturační adresa - stát
- `shippingStreet` - doručovací adresa - ulice
- `shippingCity` - doručovací adresa - město
- `shippingCode` - doručovací adresa - PSČ
- `shippingCountry` - doručovací adresa - stát
- `contacts` - informace o kontaktech spadajících pod firmu
- `invoices` - informace o fakturách přiřazených k firmě
- `customFields` - informace o vlastních polích

## 2.3 Seznamy

### 2.3 Seznamy

**Název komponenty mající na starosti příslušnou oblast:** `lists`

**Název identifikátoru pro účely aktualizace nebo mazání:** `idList`

**Povinné parametry pro uložení nového záznamu:** `title` (název seznamu)

#### 2.3.1 Struktura datové části requestu pro vložení nebo aktualizaci záznamu

- `title` - název seznamu
- `body` - popis seznamu
- `important` - příznak, zda se jedná o důležitý seznam ([0 | 1])
- `smartEmailing` - příznak, zda má být seznam synchronizován s aplikací SmartEmailing ([0 | 1])
- `customFields` - vlastní pole
  - `id => value` (ID vlastního pole => hodnota vlastního pole)
  - v případě vlastního pole typu `selectbox` lze do hodnoty vlastního pole vložit buď ID položky seznamy (doporučeno), nebo skutečnou hodnotu

#### 2.3.2 Struktura vrácených informací o záznamu

- `idList` - ID seznamu
- `title` - název seznamu
- `body` - popis seznamu
- `important` - příznak, zda se jedná o důležitý seznam ([0 | 1])
- `smartEmailing` - příznak, zda má být seznam synchronizován s aplikací SmartEmailing ([0 | 1])
- `customFields` - informace o vlastních polích

## 2.4 Aktivita

### 2.4 Aktivita

**Název komponenty mající na starosti příslušnou oblast:** `activities`

**Název identifikátoru pro účely aktualizace nebo mazání:** `idActivity`

**Povinné parametry pro uložení nového záznamu:** `idContact, body`

#### 2.4.1 Struktura datové části requestu pro vložení nebo aktualizaci záznamu

- `title` - titulek aktivity (pokud není zadán, je vytvořen automaticky z prvních 255 znaků prvního řádku (oddělovač `\n`) textu aktivity)
- `body` - text aktivity
- `idContact` - ID kontaktu, ke kterému se aktivita váže
- `type` - typ aktivity
  - `note` - poznámka
  - `call` - telefonní hovor
  - `meeting` - schůzka
  - `email` - email
  - `clicked link in autoresponder` - kliknutí na odkaz v autoresponderu
  - `clicked link in newsletter` - kliknutí na odkaz v newsletteru
  - `clicked link in trigger` - kliknutí na odkaz v trigger emailu
  - `opened autoresponder` - otevřený autoresponder
  - `opened newsletter` - otevřený newsletter
  - `opened trigger` - otevřený trigger email
  - `sent autoresponder` - odeslaný autoresponder
  - `sent newsletter` - odeslaný newsletter
  - `sent trigger` - odeslaný trigger email
  - `SMS` - SMS zpráva
- `timestamp` - časové razítko aktivity
- `duration` - čas strávený nad aktivitou
- `customFields` - vlastní pole
  - `id => value` (ID vlastního pole => hodnota vlastního pole)
  - v případě vlastního pole typu `selectbox` lze do hodnoty vlastního pole vložit buď ID položky seznamy (doporučeno), nebo skutečnou hodnotu

#### 2.4.2 Struktura vrácených informací o záznamu

- `idActivity` - ID aktivity

## 2.4 Aktivity

- `title` - titulek aktivity
- `body` - text aktivity
- `type` - typ aktivity
- `timestamp` - časové razítko aktivity
- `duration` - čas strávený nad aktivitou
- `contact` - [informace o kontaktu](#)
- `customFields` - informace o vlastních polích

## 2.5 Obchodní případy

### 2.5 Obchodní případy

**Název komponenty mající na starosti příslušnou oblast:** deals

**Název identifikátoru pro účely aktualizace nebo mazání:** idDeal

**Povinné parametry pro uložení nového záznamu:** idContact, title

#### 2.5.1 Struktura datové části requestu pro vložení nebo aktualizaci záznamu

- title - název obchodního případu
- body - popis obchodního případu
- idContact - ID kontaktu, ke kterému se obchodní případ váže  
*nebo*
- contactIds - ID kontaktů, ke kterým se obchodní případ váže (pole ID)
- idOwner - ID vlastníka obchodního případu
- rating - priorita obchodního případu (low, normal, high)
- status - stav obchodního případu (open, postponed, won, closed)
- deadline - termín (ve formátu YYYY-MM-DD)
- completedDate - datum vyhotovení (ve formátu YYYY-MM-DD)
- amount - částka
- customFields - vlastní pole
  - id => value (ID vlastního pole => hodnota vlastního pole)
  - v případě vlastního pole typu selectbox lze do hodnoty vlastního pole vložit buď ID položky seznamy (doporučeno), nebo skutečnou hodnotu

#### 2.5.2 Struktura vrácených informací o záznamu

- idDeal - ID obchodního případu
- body - popis obchodního případu
- rating - priorita obchodního případu (low, normal, high)
- status - stav obchodního případu (open, postponed, won, closed)
- deadline - termín (ve formátu YYYY-MM-DD)
- completedDate - datum vyhotovení (ve formátu YYYY-MM-DD)
- amount - částka
- contactIds - ID kontaktů, ke kterým se obchodní případ váže
- idOwner - ID vlastníka obchodního případu
- customFields - informace o vlastních polích



## 2.5 Obchodní případy

### 2.5.3 Šablony obchodních případů

API šablon obchodních případů slouží k získání informací o stávajících šablonách a vytvoření obchodního případu ze šablony, neumožňuje tedy aktuálně vkládat, upravovat či mazat stávající šablony.

**Název komponenty mající na starosti příslušnou oblast:** `deals`

**Název identifikátoru:** `idDealTemplate`

#### Získání všech šablon

Metoda: `getAllTemplates`

#### Získání šablony dle ID

Metoda: `getTemplate`

#### Vytvoření obchodního případu ze šablony

Metoda: `createFromTemplate`

**Povinné parametry pro uložení nového záznamu:** `idDealTemplate`

**Volitelné parametry:** `idContact` - ID souvisejícího kontaktu

## 2.6 Úkoly

### 2.6 Úkoly

**Název komponenty mající na starosti příslušnou oblast:** `tasks`

**Název identifikátoru pro účely aktualizace nebo mazání:** `idTask`

**Povinné parametry pro uložení nového záznamu:** `body`

#### 2.6.1 Struktura datové části requestu pro vložení nebo aktualizaci záznamu

- `title` - titulek aktivity (pokud není zadán, je vytvořen automaticky z prvních 255 znaků prvního řádku (oddělovač `\n`) textu úkolu)
- `body` - popis úkolu
- `idContact` - ID kontaktu, ke kterému se obchodní příležitost váže
- `idAssignedUser` - ID uživatele přiděleného k úkolu
- `priority` - priorita obchodní příležitosti (`very low`, `low`, `normal`, `high`, `very high`)
- `status` - stav obchodní příležitosti (`open`, `postponed`, `won`, `closed`)
- `deadline` - termín (ve formátu `YYYY-MM-DD`)
- `deadlineTime` - čas (ve formátu `HH:MM`)
- `completedDate` - datum vyhotovení (ve formátu `YYYY-MM-DD`)
- `duration` - délka trvání úkolu (ve formátu `HH:MM`)
- `customFields` - vlastní pole
  - `id => value` (ID vlastního pole => hodnota vlastního pole)
  - v případě vlastního pole typu `selectbox` lze do hodnoty vlastního pole vložit buď ID položky seznamy (doporučeno), nebo skutečnou hodnotu

#### 2.6.2 Struktura vrácených informací o záznamu

- `idTask` - ID úkolu
- `title` - titulek úkolu
- `body` - popis úkolu
- `priority` - priorita obchodní příležitosti (`very low`, `low`, `normal`, `high`, `very high`)
- `status` - stav obchodní příležitosti (`open`, `postponed`, `won`, `closed`)
- `deadline` - termín (ve formátu `YYYY-MM-DD`)
- `duration` - délka trvání úkolu (ve formátu `HH:MM`)
- `completedDate` - datum vyhotovení (ve formátu `YYYY-MM-DD`)
- `assignedUser` - informace o uživateli přiděleném k úkolu
  - `idUser` - ID uživatele
  - `username` - uživatelské jméno

## 2.6 Úkoly

- `contact` - [informace o kontaktu](#)
- `customFields` - informace o vlastních polích

### 2.6.3 Šablony úkolů

Tak jako v případě API šablon obchodních případů, také API šablon úkolů slouží k získání informací o stávajících šablonách a k vytvoření úkolu ze šablony. Neumožňuje tedy vkládat, upravovat či mazat stávající šablony.

**Název komponenty mající na starosti příslušnou oblast:** `tasks`

**Název identifikátoru:** `idTaskTemplate`

#### Získání všech šablon

Metoda: `getAllTemplates`

#### Získání šablony dle ID

Metoda: `getTemplate`

#### Vytvoření úkolu ze šablony

Metoda: `createFromTemplate`

**Povinné parametry pro uložení nového záznamu:** `idTaskTemplate`

**Volitelné parametry:**

- `idContact` - ID souvisejícího kontaktu
- `idDeal` - ID souvisejícího obchodního případu

## 2.7 Uživatelé

### 2.7 Uživatelé

#### 2.7.1 Získání seznamu aktivních uživatelů

Název komponenty mající na starosti příslušnou oblast: `users`

Název metody: `getForSelect`

#### Struktura vrácených informací

- ID uživatele => uživatelské jméno

Název metody: `getActiveUsers`

#### Struktura vrácených informací

- `idUser` – ID uživatele
- `username` – uživatelské jméno
- `email`

## 2.8 Vlastní pole

## 2.8 Vlastní pole

### 2.8.1 Získání seznamu vlastních polí

**Název komponenty mající na starosti příslušnou oblast:** příslušná komponenta obsahu

- `activities` - aktivity
- `contacts` - kontakty
- `deals` - obchodní případy
- `lists` - seznamy
- `organizations` - firmy
- `tasks` - úkoly

**Název metody:** `getCustomFields`

### Struktura vrácených informací

- `idCustomField` - ID
- `title` - název vlastní pole
- `options` - volby u vlastního pole typu `selectbox`
  - `key` - číslo volby
  - `title` - název volby
- `type` - typ vlastního pole
  - `text` - textové pole
  - `textarea` - textová oblast
  - `date` - datum
  - `number` - číslo
  - `checkbox` - zaškrtačací pole
  - `selectbox` - výběrové pole
- `weight` - váha vlastního pole (používá se pro řazení; těžší položky se propadají dolů)
- `nodeType` - typ obsahu

### 2.9 Webhooky

V některých případech může být důležité, aby systém napojený na Anabix věděl o nových záznamech či změnách co nejdříve.

Toho je možné docílit buď pravidelným voláním API Anabixu ze strany napojené aplikace, kdy pomocí metody `getAll` s vhodnou filtrací a řazením zjistíme nově změněné záznamy, a nebo použitím webhooků.

Webhooky umožňují, aby ihned po vložení či změně záznamu v Anabixu byla informace o záznamu poslána ve tvaru JSON na specifickou URL adresu na vašem serveru. Toto je realizováno pomocí triggerů, kdy lze upozornění webhooku nastavit jako tzv. *Následnou akci*. Parametrem této akce je URL, kterou chcete automaticky zavolat poté, co dojde ke *spouštěcí akci*. Na URL se zasílá POST request, který obsahuje JSON data ve struktuře:

- `idTrigger` - ID triggeru, na základě, kterého byl request zaslán
- `eventOptions` - nastavení spouštěcí události; vnitřek tohoto záznamu se liší podle nastavené události
- `actionOptions` - nastavení následné akce
  - ◆ `webhookUrl` - URL adresa, kam se request zasílá
  - ◆ `title` - "NotifyWebhook"
- `eventData` - informace o konkrétní události, na základě které došlo k sepnutí triggeru
  - ◆ `idNode` - ID záznamu, kterého se událost týká
  - ◆ `hookData` - případná doplňující data ke spouštěcí události

#### 2.9.1 Získání informací na straně cílové URL v PHP

```
<?php
$request = file_get_contents('php://input');
$data = json_decode($request, true);
var_dump($data);
?>
```

## 3 Často kladené otázky (FAQ)

### 1. Je možné získat z API data o kontaktu, když nevím jeho ID, ale jeho e-mail?

Ano, tuto informaci získáte voláním metody `contacts::getAll()`; do pole `data` vložíte vyhledávací podmínku `{"criteria":{"email":"john@doe.com"}}`. Získáte tak informace o všech kontaktech s touto emailovou adresou (systém umožňuje vložit více kontaktů se stejným emailem).

### 2. Je možné vyhledávat kontakty podle více emailů?

Ne, tuto možnost API aktuálně nepodporuje.