

Praetor GraphQL API

GraphQL – API

Příklad adresy: <https://advokatni-kancelar.cz/api/v1/graphql>

GraphiQL – webové uživatelské rozhraní ke GraphQL API

Příklad adresy: <https://advokatni-kancelar.cz/api/v1>

V tomto rozhraní lze pomocí levého panelu vytvořit dotaz a rovnou ho na serveru spustit a vidět reálnou odpověď serveru.



Nahrání dokumentu na server

Příklad adresy: <https://advokatni-kancelar.cz/api/v1/files/upload>

Aby bylo možné spustit akci „addSpisDocument“ je potřeba nahrát soubor na server a získat tak jeho ID. Samotný upload se provádí přes HTTP požadavek (viz. následující příklad).

Potřebné ID je vráceno jako hodnota `tmpFileId`.

Nahráný soubor lze použít v `addSpisDocument` jen jednou.

Příklad nahrávání dokumentu

Požadavek

```
POST /api/v1/files/upload HTTP/1.1
Content-Type: application/octet-stream
authenticationToken: YOUR_ACCESS_TOKEN
Host: advokatni-kancelar.cz
Content-Length: 3721695
...BINARY_DATA...
```

Odpověď

```
HTTP/1.1 200 OK
```

```
Transfer-Encoding: chunked
Content-Type: application/json; charset=utf-8
Date: Tue, 26 Jan 2021 16:46:21 GMT

{
  "datum": "26.01.2021 17:46:21",
  "tmpFileId": "1bb06f22-ceaf-4a43-a2e1-c84a451b2a01",
  "fileSize": "3721695"
}
```

Příklad kódu v C# pro nahrání dokumentu na server

```
string url = "https://advokatni-kancelar.cz/api/v1/files/upload";
string secToken = "YOUR_ACCESS_TOKEN";
string filePath = @"c:\tmp\image.jpg";

HttpWebRequest webRequest =
    (HttpWebRequest)WebRequest.Create(url);
webRequest.Method = "POST";
webRequest.ContentType = "application/octet-stream";
webRequest.Headers.Add("authenticationToken", secToken);

byte[] byteArray = File.ReadAllBytes(filePath);
webRequest.ContentLength = byteArray.Length;
using (Stream requestStream = webRequest.GetRequestStream())
{
    requestStream.Write(byteArray, 0, byteArray.Length);
}

using (WebResponse response = webRequest.GetResponse())
{
    using (Stream responseStream = response.GetResponseStream())
    {
        StreamReader rdr = new StreamReader(responseStream,
            Encoding.UTF8);
        return rdr.ReadToEnd();
    }
}
```

Načítání dat

Načítání dat se provádí spuštěním dotazu MyQuery, kde se definuje typ data a konkrétní data. Například následující dotaz načte a zobrazí seznam hodnot z číselníku "procesní stav" (konkrétně hodnoty Id a Název):

```
1 query MyQuery {
2   lists {
3     caseProcessStages {
4       nodes {
5         id
6         name
7       }
8     }
9   }
10 }
11
```

A výsledek po spuštění výše uvedeného dotazu:

```
{
  "data": {
    "lists": {
      "caseProcessStages": {
        "nodes": [
          {
            "id": "717e3728-55b1-472a-9729-00880931da09",
            "name": "Příprava dokumentace"
          },
          {
            "id": "35f713f0-4faa-41bc-98f9-251bd3ed27b3",
            "name": "Podána předžalobní výzva"
          },
          {
            "id": "bc7c33d2-213e-461a-9705-2d4d75ac3080",
            "name": "Podána žaloba"
          },
          {
            "id": "e0ca8e3c-6925-4e5d-a06d-33837cd29d93",
            "name": "Podání obžaloby"
          },
          {
            "id": "036e6a7b-aae9-4f0a-907f-e49a18925e1e",
            "name": "Obdržení připomínek"
          },
          {
            "id": "42a3ac77-a014-4674-b732-f798c35ed198",
            "name": "Dokumentace odeslána na klienta"
          }
        ]
      }
    }
  }
}
```

V MyQuery -> lists jsou všechny dostupné číselníky, jako například již zmiňovaný "procesní stav" nebo "Kategorie spisu", "Role subjektu", apod.

Kromě číselníků je možné načítat také přehledy, např. přehled spisů, přehled subjektů, apod. Následující ukázkový dotaz načte a zobrazí všechny spisy (konkrétně jejich Id, Předmět a Datum založení):

```
1 query MyQuery {
2   cases {
3     nodes {
4       name
5       id
6       openingDate
7     }
8   }
9 }
10
```

A výsledek po spuštění výše uvedeného dotazu:

```
{
  "data": {
    "cases": {
      "nodes": [
        {
          "name": "Výpověď nájení smlouvy",
          "id": "d84e8342-9b74-4473-8045-0454fccc77fb",
          "openingDate": "2016-01-07"
        },
        {
          "name": "Kupní smlouva - pozemek Praha",
          "id": "a9e6e30c-3a7d-47e8-8f1d-0f98a0c2d093",
          "openingDate": "2016-03-11"
        },
        {
          "name": "Odpor proti platebnímu rozkazu",
          "id": "7fb3fe3d-93f8-45bb-8f61-191af0340d4c",
          "openingDate": "2016-04-15"
        },
        {
          "name": "Dědické řízení ",
          "id": "d419bc13-4c0b-45f6-a8aa-ff1ea7d6c9ca",
          "openingDate": "2016-03-11"
        }
      ]
    }
  }
}
```

Načítání filtrovaných dat

Kromě samotného načítání všech dat je možné také filtrovat data podle některých parametrů. V případě číselníků (MyQuery -> lists) se filtruje zpravidla podle názvu (viz. následující ukázka). Volání je podobné jako při načítání všech dat, ale navíc se doplní filtrační podmínka (žlutě zvýrazněné na následujícím obrázku):

```
1 query MyQuery {
2   lists {
3     caseProcessStages(name: "Podána žaloba") {
4       nodes {
5         id
6         name
7       }
8     }
9   }
10 }
11
```

A výsledek po spuštění výše uvedeného dotazu:

```
{
  "data": {
    "lists": {
      "caseProcessStages": {
        "nodes": [
          {
            "id": "bc7c33d2-213e-461a-9705-2d4d75ac3080",
            "name": "Podána žaloba"
          }
        ]
      }
    }
  }
}
```

V případě hlavních přehledů je vyhledání konkrétního filtru již jako samostatná akce kdy namísto množného čísla "cases" je pouze "case" a a filtruje se zpravidla podle Id. Například vyhledání konkrétního spisu podle Id se provede následovně:

```
1 query MyQuery {
2   case(id: "d84e8342-9b74-4473-8045-0454fccc77fb") {
3     name
4     closingDate
5     openingDate
6   }
7 }
8
```

A výsledek po spuštění výše uvedeného dotazu:

```
{
  "data": {
    "case": {
      "name": "Výpověď nájení smlouvy",
      "closingDate": "2016-03-31",
      "openingDate": "2016-01-07"
    }
  }
}
```

Vytváření dat

Pomocí Praetor GraphQL API je možné nejen data načítat, ale také vytvářet a ukládat nové záznamy. Vytváření dat se provádí pomocí MyMutation (ve webovém uživatelském rozhraní se zapíná vlevo dole). V následujícím příkladu si ukážeme jak založit nový spis, nový subjekt a následně tento subjekt přiřadit na spis jako “zúčastněný subjekt”

V následujícím příkladu si vytvoříme nový spis. Ještě předtím si ale budeme muset zjistit některé informace (Id hlavního klienta, Id pobočky, Id stavu, Id kategorie a Id procesního stavu ... hodnoty odpovídající jednotlivým IDčkám jsme pro názornost vypsali a graficky zeleně zvýraznili)

```
1 mutation MyMutation {
2   caseCreate(
3     input: {
4       name: "Testovací spis",
5       note: "Testovací poznámka",
6       mainClientId: "551B1021-44BC-4B65-0001-000000000640", // "Karel Novák"
7       caseBranchId: "cdb3a93b-a8fc-48c7-b03c-834190a7830c", // "Brno"
8       caseStateId: "3b73ccc2-0c94-42a6-a434-4497f4d57e75", // "Aktivní"
9       caseCategoryId: "9a8abb53-9dc1-4ca9-bcc4-a652e92ed402", // "Trestní agenda"
10      caseProcessStageId: "bc55c9f0-1ea3-4ca0-a054-9249a1427961", // "Vyúčtování"
11      openingDate: "1.3.2021"
12    }
13  )
14  {
15    case {
16      id
17      name
18    }
19  }
20 }
```

A výsledek po spuštění výše uvedeného dotazu:

```
{
  "data": {
    "caseCreate": {
      "case": {
        "id": "4751da30-4e60-420c-a6cf-1dcb9c05f79f",
        "name": "Testovací spis"
      }
    }
  }
}
```

Z výsledku je zřejmé, že se spis do databáze úspěšně uložil a jeho Id je hodnota "4751da30-4e60-420c-a6cf-1dcb9c05f79f" (tuto hodnotu budeme ještě následně potřebovat abychom na tento spis mohli navázat další údaje, případně provést update tohoto spisu).

Nyní si (podobně jako spis) vytvoříme ještě subjekt:

```
1 mutation MyMutation {
2   personCreate(
3     input: {
4       firstName: "Šárka",
5       surname: "Nováková",
6       note: "Testovací osoba",
7       personTypeId: "1" // "Fyzická osoba - nepodnikatel"
8     }
9   )
10  {
11    person {
12      id
13    }
14  }
15 }
16
```

A výsledek po spuštění výše uvedeného dotazu:

```
{
  "data": {
    "personCreate": {
      "person": {
        "id": "43a7b703-a7ae-4bd3-ae0-2b53b2fd3f1a"
      }
    }
  }
}
```

Z výsledku je opět patrné, že se subjekt do databáze úspěšně uložil a jeho Id je hodnota "43a7b703-a7ae-4bd3-ae0-2b53b2fd3f1a" (tuto hodnotu budeme opět ještě následně potřebovat, abychom tuto osobu mohli přiřadit ke spisu).

Všimněte si, že oproti tentokrát byla návratová hodnota pouze Id (ačkoliv při zakládání spisu se vrátilo Id i Předmět). Jaká data mají být součástí návratové hodnoty si můžete sami zvolit, v tomto našem konkrétním případě jsme si požádali o vrácení pouze ID (viz. řádek 12 na obrázku s příkazem pro založení subjektu).

A nyní si námi vytvořený subjekt (Šárku Novákovou) přiřadíme na náš spis jako “zúčastněný subjekt”:

```
1 mutation MyMutation {
2   caseAddPerson
3   (
4     input: {
5       roleId: "19", // "Zúčastněný subjekt"
6       personId: "43a7b703-a7ae-4bd3-aea0-2b53b2fd3f1a",
7       caseId: "4751da30-4e60-420c-a6cf-1dcb9c05f79f"
8     }
9   )
10  {
11    personEdge {
12      edgeInfo {
13        id
14      }
15    }
16  }
17 }
```

A výsledek po spuštění výše uvedeného dotazu:

```
{
  "data": {
    "caseAddPerson": {
      "personEdge": {
        "edgeInfo": {
          "id": "4ffb8d21-ce4c-4107-927c-1aa4ecfe78ef"
        }
      }
    }
  }
}
```

Z výsledku je opět patrné, že se uložení do DB podařilo. O jako ověření se ještě můžeme podívat přímo do Praetora:

The screenshot shows the Praetor application interface. The top navigation bar includes 'Moje agenda', 'Spisy', and '2021/003'. The main content area displays '2021/003 — Testovací spis'. On the left, a sidebar menu lists 'Základní údaje', 'Log činnosti', 'Subjekty', 'Dokumenty', 'Pošta', and 'Lhůty a úkoly'. The 'Subjekty' section is active, showing a table with columns 'R', 'Označení', and 'Identifikace'. The table lists 'Klient', 'Praetor Systems s.r.o.' (IČO: 24694380), and 'Zúčastněné subjekty'. Under 'Zúčastněné subjekty', the name 'Šárka Nováková' is visible. A white circle highlights the 'Zúčastněné subjekty' section and the name 'Šárka Nováková'.

Editace dat

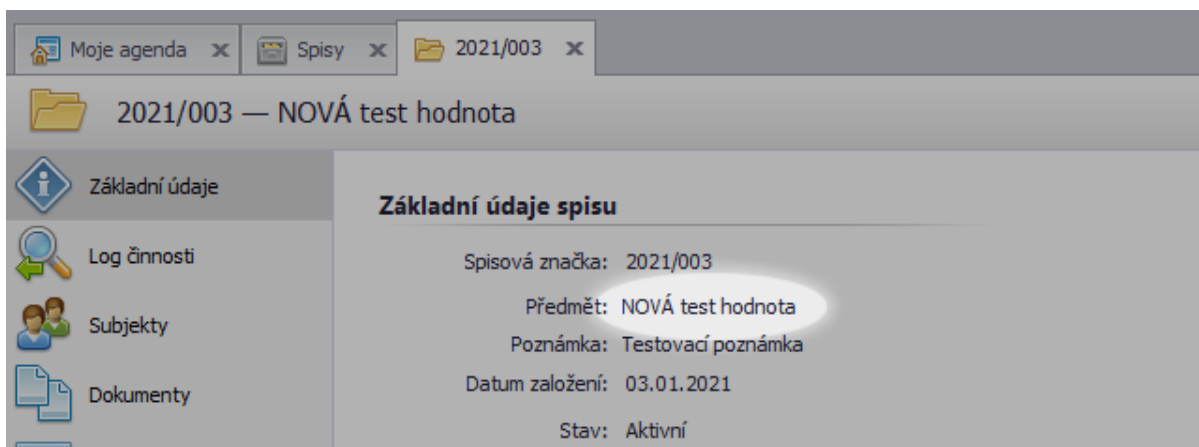
Editace (úprava) dat je velmi podobná akce jako vytváření s tím rozdílem, že je potřeba znát Id záznamu, který chceme upravit. V následující ukázce si předvedeme, jak u našeho spisu upravíme předmět:

```
1 mutation MyMutation {
2   caseUpdate(
3     input: {
4       id: "4751da30-4e60-420c-a6cf-1dcb9c05f79f",
5       name: "NOVÁ test hodnota"
6     }
7   )
8 }
9 {
10  case {
11    id
12    name
13    note
14  }
15 }
```

A výsledek po spuštění výše uvedeného dotazu:

```
{
  "data": {
    "caseUpdate": {
      "case": {
        "id": "4751da30-4e60-420c-a6cf-1dcb9c05f79f",
        "name": "NOVÁ test hodnota",
        "note": "Testovací poznámka"
      }
    }
  }
}
```

A ještě vizuální kontrola přímo v Praetoru:



The screenshot shows the Praetor application interface. At the top, there are tabs for 'Moje agenda', 'Spisy', and '2021/003'. Below the tabs, the breadcrumb path is '2021/003 — NOVÁ test hodnota'. On the left side, there is a sidebar with icons for 'Základní údaje', 'Log činnosti', 'Subjekty', and 'Dokumenty'. The main content area displays the 'Základní údaje spisu' (Basic case details) for the case with ID 2021/003. The details are as follows:

Spisová značka:	2021/003
Předmět:	NOVÁ test hodnota
Poznámka:	Testovací poznámka
Datum založení:	03.01.2021
Stav:	Aktivní

Závěrem

Pro počáteční seznámení je ideální používat webové uživatelské rozhraní, kde máte přehled aktuálně dostupných akcí a jednotlivé dotazy si můžete jednoduše naklikat.

Příklad kódu v C# pro odesílání dotazů na server

V následujícím kusu kódu si ukážeme, dotazy (ať už pro načítání, tak i pro vytváření či úpravu dat) spouštět z C# aplikace:

```
var prikaz = "query{ cases { nodes { id } } }";

var token = "YOUR_ACCESS_TOKEN";
var url = "https://advokatni-kancelar.cz/api/v1/graphql";

var hc = new HttpClient();
var query = "{\"query\":\"" + JsonConvert.ToString(prikaz) + "\"}";
var byteArray = Encoding.UTF8.GetBytes(query);
var content = new ByteArrayContent(byteArray);
content.Headers.ContentType = new
MediaTypeHeaderValue("application/json");
content.Headers.Add("authenticationToken", token);
var response = await hc.PostAsync(url, content);
var result = await response.Content.ReadAsStringAsync();
```