Absinthe

The GraphQL toolkit for Elixir

Jan Zborowski

Programowanie w językach Erlang i Elixir 2020/2021

Ale co to jest GraphQL?

Język zapytań do API

+

System typów i język definujący strukturę danych udostępnianych przez API

+

Specyfikacja określająca jak obsługiwać zapytania o zdefiniowane dane

1. Query - do odczytu danych

```
query {
  stations {
    name
    coords {
      latitude
      longitude
    }
  }
}
```

```
{
  "data": {
    "stations": [{
        "coords": {
          "latitude": 41,
          "longitude": 20
        },
        "name": "Foo"
     }]
}}
```

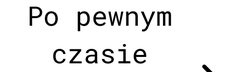
2. Mutation - do zapisu danych

```
mutation {
  addStation(name: “Foo”,
             coords: {latitude: 41, longitude: 20}) {
    name
    coords {
      latitude
      longitude
}}}
```

```
{
  "data": {
    "addStation": {
      "coords": {
        "latitude": 41,
        "longitude": 20
      },
      "name": "Foo"
    }
}}
```

3. Subscription - do asynchronicznego odczytu danych związanych ze zdarzeniami

```
subscription {
  stationAdded {
    name
  }
}
```




```
{
  "data": {
    "stationAdded": {
      "name": "Foo"
    }
  }
}
```

Rodzaje zapytań

Struktura danych (Schema)

Typy skalarne + Obiekty

```
Int, Float, String,
Boolean, ID
```

```
type SomeObject {
  foo: Int,
  bar: SomeOtherObject
}
```

```
type Coords {
  latitude: Float!
  longitude: Float!
}
```

```
input CoordsInput {
  latitude: Float!
  longitude: Float!
}
```

```
type Station {
  coords: Coords!
  name: String!
}
```

```
type Query {
  stations: [Station!]!
}
```

```
type Mutation {
  addStation(
    name: String!,
    coords: CoordsInput!
  ): Station!
}
```

```
type Subscription {
  stationAdded: Station!
}
```

1. Implementuje silnik obsługi zapytań GQL

Parsuje, waliduje i wykonuje zapytania GraphQL

2. Umożliwia zdefiniowanie struktury danych(Schema) w Elixirze

Dostarcza zestaw deklaratywnych makr i funkcji pozwalających na definiowanie struktury danych w idiomatyczny sposób

3. Integruje się z istniejącymi narzędziami ekosystemu Elixira i klienckimi

Oprócz core'owej paczki *absinthe*, mamy *absinthe_plug*, *absinthe_phoenix*, *dataloader*, oraz wsparcie dla frontendowych bibliotek GraphQL*: Relay i Apollo.*

A co robi Absinthe?

Demo

https://github.com/john-sonz/pollution-absinthe-example

Absinthe 

- Absinthe pozwala tworzyć GraphQL'owe API w idiomatycznym Elixirze
- Dobrze integruje się z innymi narzędziami jak Phoenix i Ecto.
- Jest stale rozwijany i wspierany przez Elixirową społeczność

Materiały dla zainteresowanych

Absinthe

GraphQL

How to GraphQL + Absinthe

Szablon prezentacji

Podsumowanie