

# The Graph

如何提高  
Dapp  
开发效率



什么是Web3?

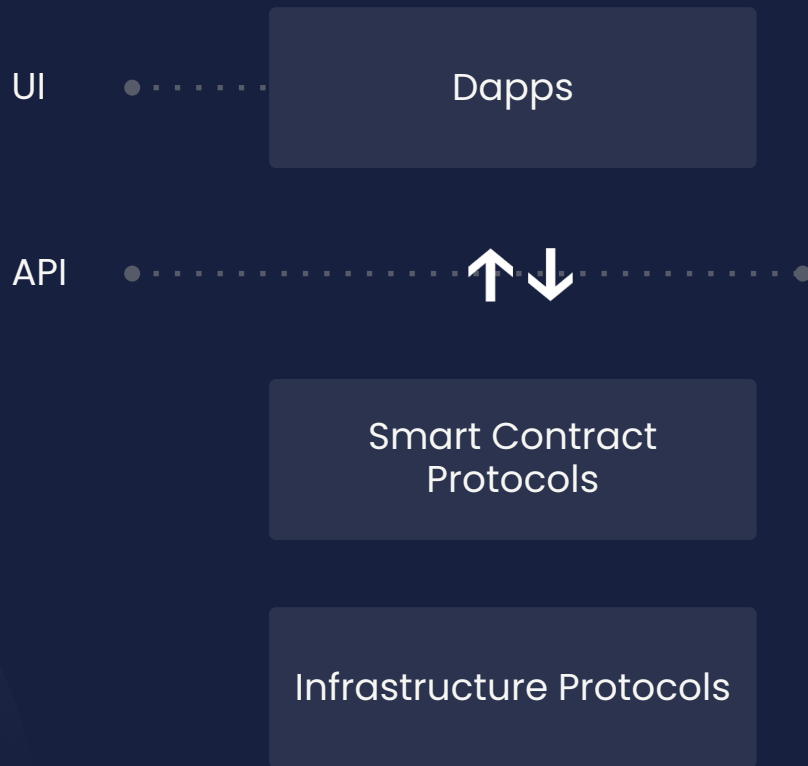
# 构建分布式应用的 新平台



Web3  
无服务器



# 什么是 Web3?



Web3

# 基础设施



Onchain

Offchain

Bridges

L2s

Textile

Ceramic

Ethereum

Other L1s

Notifications

IPFS

Storage

Video

Web3

# 搜索与查询

没有The Graph

## Blocks

The latest block height is #51769080. Refresh or Click to view the latest data.

#51769076	Finalized 11s ago
#51769075	Finalized 12s ago
#51769074	Finalized 13s ago
#51769073	Finalized 15s ago
#51769072	Finalized 16s ago
#51769071	Finalized 17s ago
#51769070	Finalized 18s ago
#51769069	Finalized 19s ago
#51769068	Finalized 20s ago

Web3

# 搜索与查询

没有The Graph

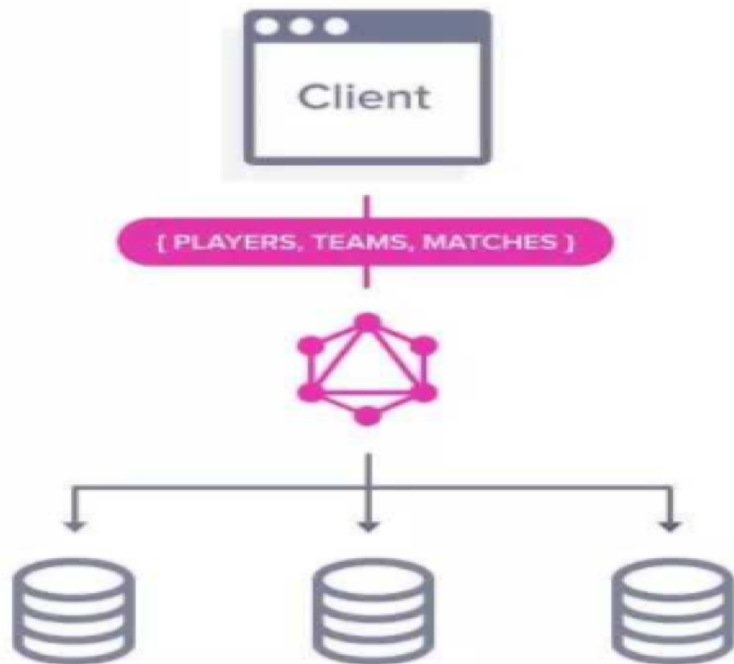


Web3

# 搜索与查询

有了The Graph

## GraphQL API





Web3

# 搜索与查询

有了The Graph

```
{  
  hero {  
    name  
  }  
}
```

```
{  
  "hero": {  
    "name": "Luke Skywalker"  
  }  
}
```

# 查询架构



Client

Proof



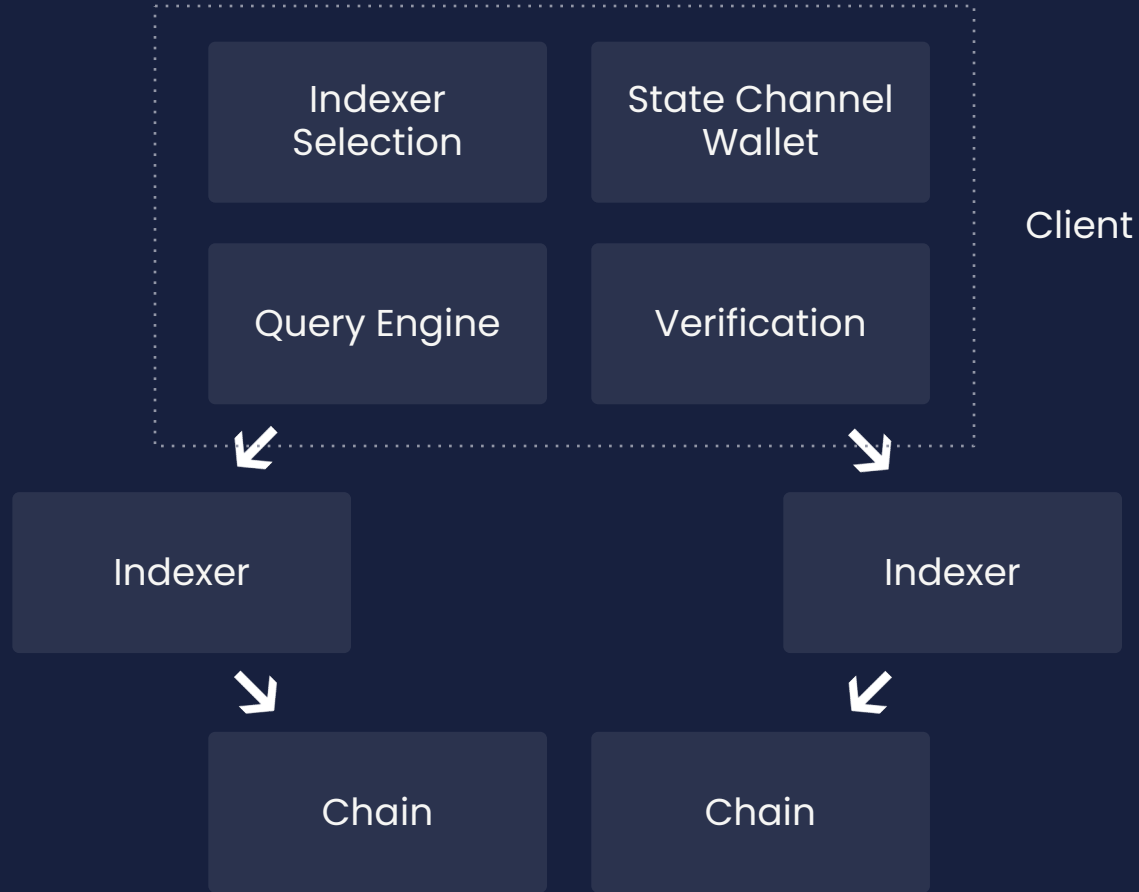
Indexers

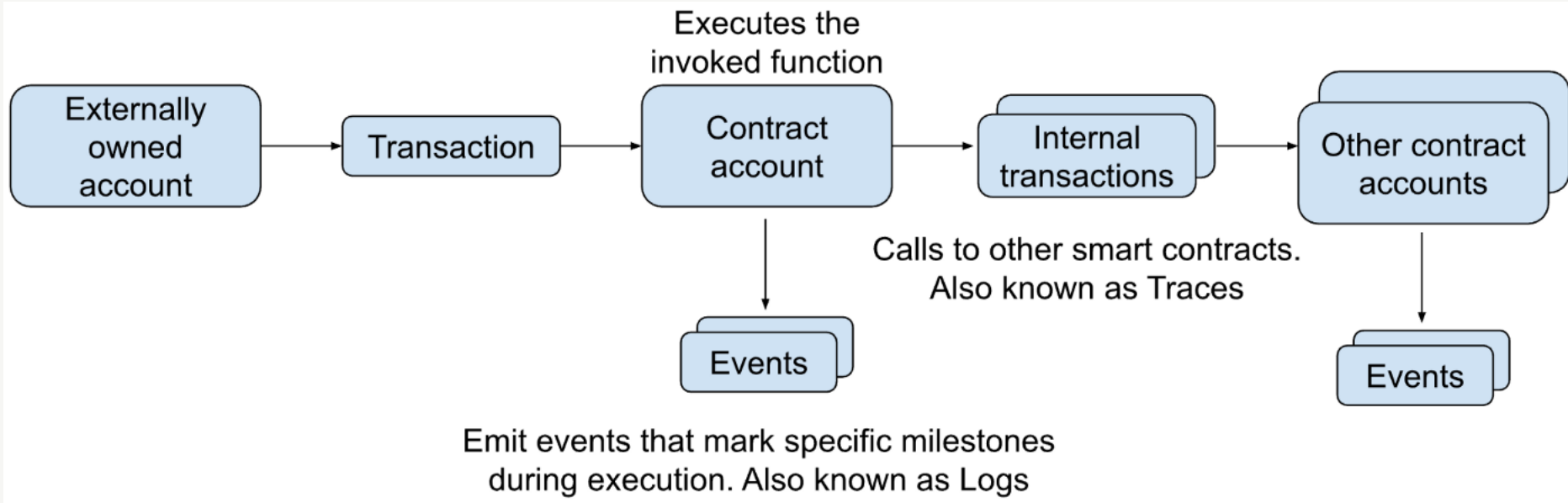


Blockchains

Storage  
Networks

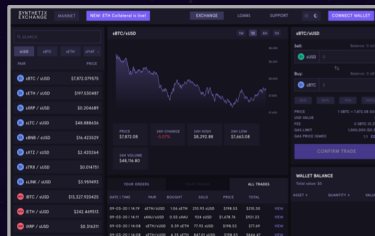
# Graph Client



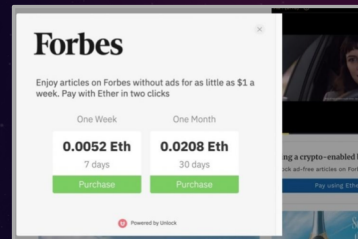


# 支持各种复杂的去中心化应用

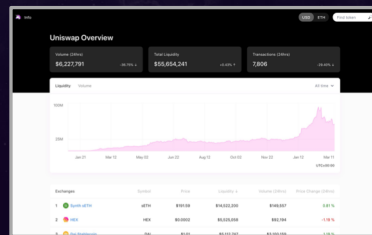
## 交易数据



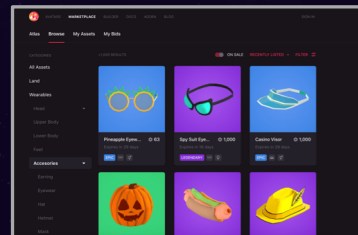
## Forbes 订阅数据



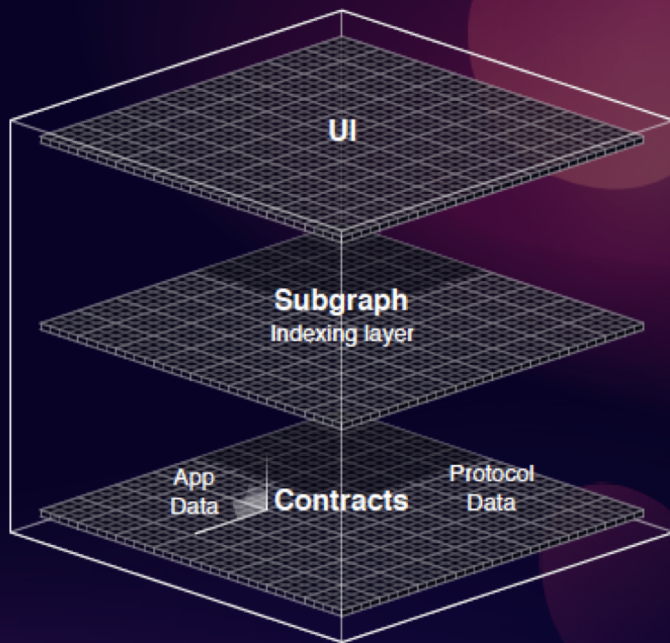
## 去中心化交易平台



## 虚拟资产

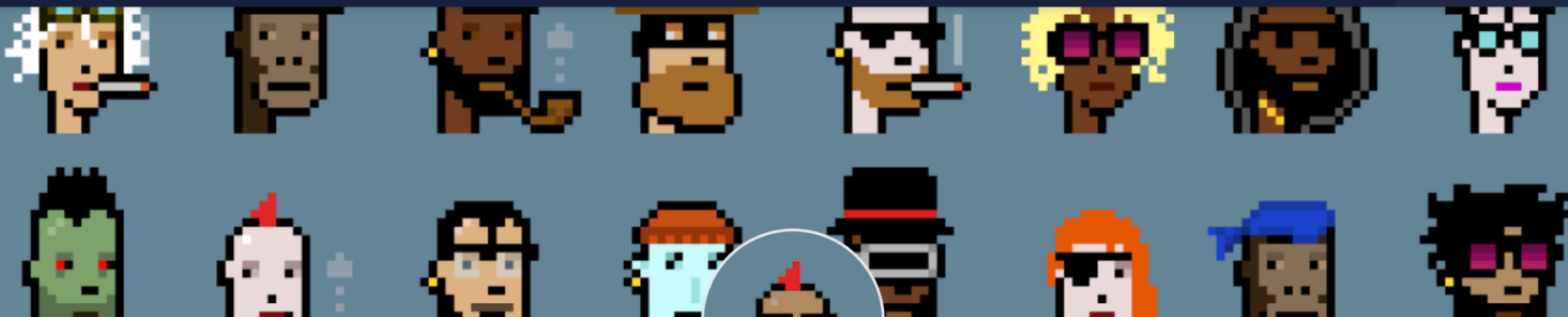


# 成为web3技术栈 不可或缺的基础 设施



如何构建Subgraph?

# 分布式应用



+ Add to watchlist



## CryptoPunks

10.0K

items

3.4K

owners

◆ ---

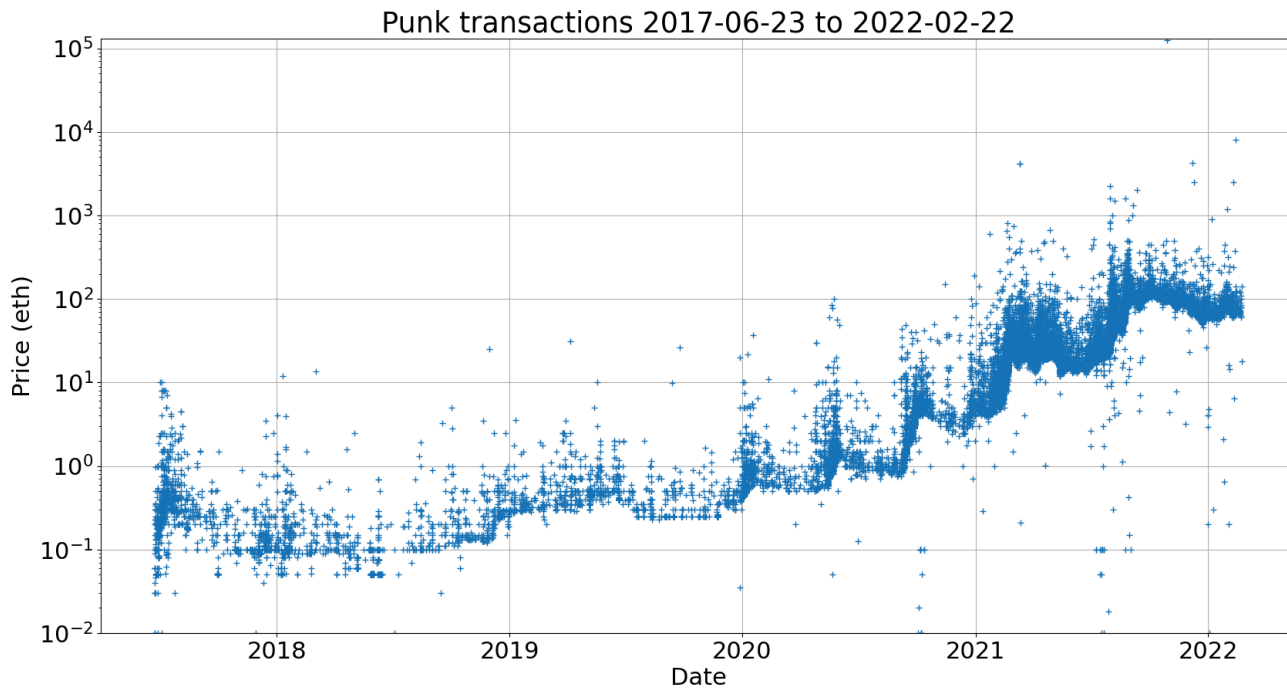
floor price

◆ 845.0K

volume traded

如何构建Subgraph?

# 分布式应用





如何构建Subgraph?

# 工具

[@graphprotocol/graph-cli](#)

[@graphprotocol/graph-ts](#)



如何构建Subgraph?

# 命令

\$ graph codegen

\$ graph build



如何构建Subgraph?

# 代码

subgraph.yaml

schema.graphql

AssemblyScript映射



# subgraph.yaml

```
1 specVersion: 0.0.2
2 schema:
3   file: ./schema.graphql
4 #graft:
5 # base: Qmf9kah4dfYmaEBRriqXEuHXMGRUCKNHcytYSasCygAF9K
6 # block: 10821737
7 dataSources:
8   - kind: ethereum/contract
9     name: cryptopunks
10    network: mainnet
11    source:
12      address: "0xb47e3cd837dDF8e4c57F05d70Ab865de6e193BBB"
13      abi: cryptopunks
14      startBlock: 3914495
15    mapping:
16      kind: ethereum/events
17      apiVersion: 0.0.5
18      language: wasm/assemblyscript
19      entities:
20        - Account
21        - Punk
22        - NFT
23        - Transfer
24        - MetaData
25        - Trait
26        - WrappedPunk
27        - Assign
28        - Offer
29        - Transfer
30        - Sale
31      abis:
32        - name: cryptopunks
33          file: ./abis/cryptopunks.json
34        - name: CryptopunksMetadata
35          file: ./abis/CryptopunksMetadata.json
```

# subgraph.yaml

```
36     eventHandlers:
37       - event: Assign(indexed address,uint256)
38         handler: handleAssign
39       - event: PunkTransfer(indexed address,indexed address,uint256)
40         handler: handlePunkTransfer
41       - event: PunkOffered(indexed uint256,uint256,indexed address)
42         handler: handlePunkOffered
43       - event: PunkBidEntered(indexed uint256,uint256,indexed address)
44         handler: handlePunkBidEntered
45       - event: PunkBidWithdrawn(indexed uint256,uint256,indexed address)
46         handler: handlePunkBidWithdrawn
47       - event: PunkBought(indexed uint256,uint256,indexed address,indexed address)
48         handler: handlePunkBought
49       - event: PunkNoLongerForSale(indexed uint256)
50         handler: handlePunkNoLongerForSale
51     file: ./src/mapping.ts
```

# Schema

```
45 type Account @entity {
46   "Ethereum Address"
47   id: ID!
48
49   "All Punks owned by Account"
50   punksOwned: [Punk!] @derivedFrom(field: "owner")
51
52   "Purchases by Account"
53   bought: [Sale!]! @derivedFrom(field: "to")
54
55   "All Punks owned by Account"
56   nftsOwned: [NFT!]! @derivedFrom(field: "owner")
57
58   "Punks assigned to account (if any)"
59   assigned: [Assign!]! @derivedFrom(field: "to")
60
61   "Punk transfer by Account"
62   sent: [Transfer!]! @derivedFrom(field: "from")
63
64   received: [Transfer!]! @derivedFrom(field: "to")
65
66   "Query bids to Account or by Account"
67   bids: [Bid!]! @derivedFrom(field: "from")
68
69   "Punks offered for sale by Account"
70   asks: [Ask!]! @derivedFrom(field: "from")
71
72   numberOfPunksOwned: BigInt!
73 }
```

## 映射文件

```
export function handlePunkBought(event: PunkBought): void {
  log.debug("handlePunkBought", []);

  let sale = getOrCreateSale(
    event.params.toAddress,
    event.params.fromAddress,
    event.params.punkIndex,
    event
  );
  let punk = Punk.load(event.params.punkIndex.toString());
  let contract = getOrCreateCryptoPunkContract(event.address);
  let toAccount = getOrCreateAccount(event.params.toAddress);
  let fromAccount = getOrCreateAccount(event.params.fromAddress);

  sale.amount = event.params.value;

  contract.totalAmountTraded = contract.totalAmountTraded.plus(
    event.params.value
  );
  contract.totalSales = contract.totalSales.plus(BigInt.fromI32(1));
}
```

## 映射文件

```
// Note: buyPunk() does not emit a PunkTransfer event, so we need to keep track
toAccount.numberOfPunksOwned = toAccount.numberOfPunksOwned.plus(
    BigInt.fromI32(1)
);

fromAccount.numberOfPunksOwned = fromAccount.numberOfPunksOwned.minus(
    BigInt.fromI32(1)
);












punk.purchasedBy = toAccount.id;
punk.owner = toAccount.id;

punk.save();
fromAccount.save();
toAccount.save();
contract.save();
sale.save();
}
```



## 完整代码

<https://github.com/itsjerryokolo/CryptoPunks>

 abis	metadata contract
 contracts	Add CryptoPunks contract
 src	Cleanup test & mappings
 tests	Cleanup test & mappings
 .gitignore	Create helper function for Punk Entity, Cleanup Assign helper function
 README.md	Draft of organizing wrapped punk transfers/wrap/unwrap
 docker-compose.yml	Add docker-compose, update dependencies
 package.json	Upgrade graph cli, matchstick & graph-ts
 schema.graphql	Refactor blockHhash to blockHash
 subgraph.yaml	Start tracking the UserProxies
 yarn.lock	Upgrade graph cli, matchstick & graph-ts

如何在Near上构建Subgraph?

# 部署

<https://thegraph.com/hosted-service/dashboard>

## CREATE A SUBGRAPH

Step 1

### Create

Give the Subgraph a name and a cover image

Add subgraph

Step 2

### Install

Install Graph CLI globally using NPM or Yarn so you can build and deploy Subgraphs

Show commands

Step 3

### Init

Scaffold a Hello World Subgraph that indexes data off Ethereum mainnet

Show commands

Step 4

### Deploy

Deploy your subgraph to the Graph Node and publish it to our community once you are ready

Show commands

如何查询Subgraph?

# 查询

<https://thegraph.com/hosted-service/subgraph/itsjerryokolo/cryptopunks>

The screenshot displays the The Graph Playground interface. At the top, there are tabs for "Playground" and "Logs". The main area is divided into three sections:

- Query:** A text input field containing the query: 

```
Owner data
Default
accounts(where: { id: "0x94de7e2c73529ebf3206aa3459e699fbcdfcd49b" }) {
  id
  nftsOwned {
    id
  }
  bought {
    id
    timestamp
    txHash
  }
  assigned {
    id
    timestamp
    txHash
  }
}
```
- Response:** A text area showing the JSON response: 

```
{
  "data": {
    "accounts": [
      {
        "id": "0x94de7e2c73529ebf3206aa3459e699fbcdfcd49b",
        "nftsOwned": [
          {
            "id": "3376"
          },
          {
            "id": "3751"
          },
          {
            "id": "4704"
          }
        ]
      }
    ]
  }
}
```
- Schema:** A list of schema types: Trait, Ask, Bid, Contract, Assign, Sale, AskCreated, BidCreated, BidRemoved, AskRemoved, Transfer, Wrap, Unwrap, and UserProxy. There are "Schema" and "Hide schema" controls.

# 更多信息

<https://thegraph.com>



微信公众号

graphprotocol中文

Twitter

@TheGraphCN

Telegram

<https://t.me/GraphProtocolCN>

