


# 实时订阅指定代币合约的日志或事件

## 简介

本节将介绍几种日志订阅的方法。

我们将以订阅所有日志、订阅待确认交易、订阅新的区块头生成事件为例。

 和获取过去日志信息不同，使用订阅方式将会监听最新的区块数据，即通过 `subscribe` 方法订阅的日志是实时的。

## 订阅所有日志

通过调用 `web3.eth.subscribe` 方法，指定订阅类型为 `'logs'` 实现。

示例代码如下：

```
1 // 监听所有的 DAI 合约发起转账的事件
2 const subscription = web3.eth.subscribe('logs', {
3   address: '0x6B175474E89094C44Da98b954EedeAC495271d0F',
4   topics: ['0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef
5 }, function(error, result){
6   if (!error)
7     console.log('监听到的日志全部信息:');
8     console.log(result);
9 })
10 .on("connected", function(subscriptionId){
11   console.log(subscriptionId);
12 })
13 .on("data", function(log){
14   console.log('日志解析数据:');
15   var decodeData = web3.eth.abi.decodeLog(inputAbi, log.data, [log.topics[1],
16   console.log('from: ', decodeData[0]);
17   console.log('to: ', decodeData[1]);
18   console.log('value: ', decodeData[2]);
19 })
20 .on("changed", function(log){
21 });
```

在上面的脚本代码中，我们指定了订阅类型为 logs（订阅所有日志信息），然后我们通过过滤器选项对日志信息进行过滤。

- address：表示我们想要获取日志信息的代币合约地址（这里是 DAI 的合约地址）
- topics：第一个索引位置表示事件的签名哈希值，这里代表是 Transfer 事件

这里表示我们想要获取关于 DAI 合约的所有 Transfer 事件日志信息，输出如下：

```
$ node subscribe.js
订阅成功，订阅号： 0x1304f944348a3991b60726459e33875d
日志解析数据：
from: 0xBd4DBE0CB9136FFb4955ede88EBD5e92222aD09a
to: 0xA88800CD213dA5Ae406ce248380802BD53b47647
value: 1065010482024299820277
监听到的日志全部信息：
{
  address: '0x6B175474E89094C44Da98b954EedeAC495271d0F',
  topics: [
    '0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef',
    '0x000000000000000000000000bd4dbe0cb9136ffb4955ede88ebd5e92222ad09a',
    '0x000000000000000000000000a88800cd213da5ae406ce248380802bd53b47647'
  ],
  data: '0x000000000000000000000000000000000000000000000000000000000000000039bbfd4ebbd5eef4f5',
  blockNumber: 17453235,
  transactionHash: '0x5189b9776efad48a3d20a65961d0d5f0d5a81153fbad202cdfed4a127e3f944f',
  transactionIndex: 0,
  blockHash: '0x7346de53ce0f3322848a03630d2248df0a96ff0746c6285bd3da0380970ac461',
  logIndex: 2,
  removed: false,
  id: 'log_479411fe'
}
日志解析数据：
from: 0xA88800CD213dA5Ae406ce248380802BD53b47647
to: 0x997468c6C249bf2D24987FD07Dc929C29382189b
value: 1065010482024299820277
监听到的日志全部信息：
{
  address: '0x6B175474E89094C44Da98b954EedeAC495271d0F',
  topics: [
    '0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef',
    '0x000000000000000000000000a88800cd213da5ae406ce248380802bd53b47647',
    '0x000000000000000000000000997468c6c249bf2d24987fd07dc929c29382189b'
  ],
  data: '0x000000000000000000000000000000000000000000000000000000000000000039bbfd4ebbd5eef4f5',
  blockNumber: 17453235,
  transactionHash: '0x5189b9776efad48a3d20a65961d0d5f0d5a81153fbad202cdfed4a127e3f944f',
  transactionIndex: 0,
  blockHash: '0x7346de53ce0f3322848a03630d2248df0a96ff0746c6285bd3da0380970ac461',
  logIndex: 4,
  removed: false,
  id: 'log_913159eb'
}
```

将会监听所有 DAI 合约上最新发生的 Transfer 事件并打印。

## 订阅待处理交易

和订阅所有日志类似，订阅待处理交易时，只需要将参数 'logs' 替换为 'pendingTransactions'，示例代码如下：

```
1 // 监听所有的 DAI 合约发起转账的事件
2 const Web3 = require("web3");
3 const web3 = new Web3('wss://eth-mainnet.g.alchemy.com/v2/U3Qa-y48FUN_MV-L4BdGD9
4
5 const subscription = web3.eth.subscribe('pendingTransactions',
6     function(error, result){
7         if (!error)
8             console.log('待处理交易哈希值: ',result);
9     })
10 .on("data", function(transaction){
11     console.log(transaction);
12 });
13
14 // 取消订阅
15 subscription.unsubscribe(function(error, success){
16     if(success)
17         console.log('Successfully unsubscribed!');
18 });
```

## • 输出结果

```
待处理交易哈希值: 0xf6b96ea63575f72b46c272a5309a6c5dbd588fff779342a20ff6205d8a9a74f6
待处理交易哈希值: 0xa3f957028df1f74c6d066f40c067892245edf9aeb3bdadcceffff4075140e2545
待处理交易哈希值: 0x7e5fe4c42bbf4aaab8ce49bb8f7ed6b939aa87e36227639f31e25df1890056e1
待处理交易哈希值: 0xc79082cda22881a839c7196228503dd0cdaa0420d9264e7ee64350ed68781cca
待处理交易哈希值: 0x4a3d2eb5557c875e7683063e60e5c8b3deb5fc464c5ee544571a197fd497ddd8
待处理交易哈希值: 0x2d26330efa3d0ebe1be725f7deb96dcff5b9b761d45311fb2ead6dc37812ff43
待处理交易哈希值: 0xedf5e0bba4ef686f387a6827d9d854f1c93c5d305340ff8731bd449b4ac59c25
待处理交易哈希值: 0x4cfd9f146546b197eb90bf2428188e39fab6eeed0bde5ad1ab266df693dad037
待处理交易哈希值: 0x2d1708e43412913d5ccc2458014cde0b4764ade9f124028751613187f6e066eb
待处理交易哈希值: 0x2bf3df1be66233a3accc7ed026b82eea6f980025c5ce19476601cf8353820390
待处理交易哈希值: 0x14a2fa8cff23fed8b805f17163b09a695d5aef35dbf00e060873cf9adf30bb8a
待处理交易哈希值: 0x76ff9bee008661a3511afc9e776ca9a76bb381d4fb3488124172a2ad37791b6b
待处理交易哈希值: 0xfc2e0405e41b449ae2f3749ed4ba004abf12f7d6bb4b0dc4e63113ae8f490476
待处理交易哈希值: 0x6b20a270c65a95998a3319f2bb15a1a99f309984e0015c1891a1523ab5634122
待处理交易哈希值: 0x46a5849b3d5303be2ded62b46f3d44d55c490263874ced3422ccabf6aa59f684
待处理交易哈希值: 0x21e68a597e905526e4353bb0e9915730d18cbca39df115b275bf04db9520ed13
待处理交易哈希值: 0x84418c8d3c5efe32273b9db139dd110ea81d5035876191592bb239ac6097f7ba
待处理交易哈希值: 0x45d8b3a3e6b7d5e1142b0df66a970b5c55877c841796e6fffc0297f2411ca37eb
待处理交易哈希值: 0x56c73b255464f0d4d36e582e57764a1087769ba7cbb4b8869600de394ed117d0
待处理交易哈希值: 0x002df092b00a39607e2d91a41793c0b8e67f8352e81a49d31e8a75fbdf27cbe
```

## 订阅新的区块生成事件

直接贴脚本

```
1 const Web3 = require("web3");
2 const web3 = new Web3('wss://eth-mainnet.g.alchemy.com/v2/U3Qa-y48FUN_MV-L4BdGD9
3
4 var subscription = web3.eth.subscribe('newBlockHeaders', function(error, result)
5     if (!error) {
6         console.log(result);
7         return;
8     }
9
10     console.error(error);
11 })
12 .on("connected", function(subscriptionId){
13     console.log(subscriptionId);
14 })
15 .on("data", function(blockHeader){
16     console.log(blockHeader);
17 })
18 .on("error", console.error);
19
20 // 取消订阅
21 subscription.unsubscribe(function(error, success){
22     if (success) {
23         console.log('Successfully unsubscribed!');
24     }
25 });
```

输出新的区块详细信息

```
{
  baseFeePerGas: 15765488809,
  difficulty: '0',
  extraData: '0x7273796e632d6275696c6465722e78797a',
  gasLimit: 30000000,
  gasUsed: 7672491,
  hash: '0x7b8ecc78b7f4e61f9e8c8241a5520c48edb985e078c3b5334d4c4d938dcc17ac',
  logsBloom: '0x1021c0024092206c13004338daa0062900904108281511004c0950008297010010a8b70080100c090722
0a240811558120080b401880c44822183700210b505a145006181878607508009dbdd40224ae122484a239008078b8681092
0000010000300a2049444981030f144c491f59',
  miner: '0x1f9090aaE28b8a3dCeaDf281B0F12828e676c326',
  mixHash: '0xe969d3c3cc237f9837e3ca6306c9604e82bc22cfddb72e241810e70e467cb7a2',
  nonce: '0x0000000000000000',
  number: 17453308,
  parentHash: '0x5ec9d7602ee51eab127cbf8b8068784ac9af2094e16b2809f7d557da2d2784e0',
  receiptsRoot: '0x12bbff3005c017409e9d7c57c6ce4165beb225d08948d8820f53db50e8825fb3',
  sha3Uncles: '0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347',
  size: 84816,
  stateRoot: '0x5f5e1d8d8ac62709f9cfca7fdc102b3770fa0ae6da90bcc28781eb9533915b17',
  timestamp: 1686443087,
  transactionsRoot: '0x951d45d280c5f2e49b1c8baa95ab0d433e23425dd1341e17a91efef14d6e0362',
  withdrawals: [
    {
      index: '0x662660',
      validatorIndex: '0x3865f',
      address: '0xb9d7934878b5fb9610b3fe8a5e441e8fad7e293f',
      amount: '0xd265c5'
    },
    {
      index: '0x662661',
      validatorIndex: '0x38660',
      address: '0xb9d7934878b5fb9610b3fe8a5e441e8fad7e293f',
      amount: '0xd3019d'
    },
    {
      index: '0x662662',
      validatorIndex: '0x38661',
      address: '0xb9d7934878b5fb9610b3fe8a5e441e8fad7e293f',
      amount: '0xd2b90b'
    },
    {
      index: '0x662663',
      validatorIndex: '0x38662',
      address: '0xb9d7934878b5fb9610b3fe8a5e441e8fad7e293f',
      amount: '0xd34b3d'
    }
  ],
}
```