# How can we retrieve the evolution of the state of an smart contract variable?

I have a simple smart contract with a variable color which stores a string.

If I unterstand correctly, each transaction can change the state of this variable and is stored permanently into the blockchain. Let's say tx#1 changes it to red, tx#1 changes it to blue, and tx#2 to red, tx#3 to green.

I want to have red, blue and green, and not just the last state of this variable color.

How can I achieve this?

Thanks

solidity contract-development transactions blockchain ethereumjs

edited Mar 5 at 15:28

Rob Hitchens
**12.4k** 1 14 44

asked Jan 4 at 16:13

FrenchieiSverige
**623** 2 18

## 2 Answers

The usual approach is to emit events for all state changes. These can be monitored externally, including replay from the beginning and/or filters. It's good way to expose a history without using too much contract (expensive) storage.

```
bytes32 x;

event LogChangedX(bytes32 X);

function doSomething(bytes32 newX) {
  x = newX;
  LogChangedX(x);
```

Clients, e.g. browsers or servers, can explore the event log to discover all the changes that happened to "x". Uses web3 event listeners.

There's a more gas-expensive, storage-intensive approach that more closely resembles a database txn history table, but it's not needed for most use-cases.

Hope it helps.

edited Jan 25 at 11:59

answered Jan 4 at 16:27

Rob Hitchens
**12.4k** 1 14 44

It helps partially. I really need to store colors as string. I took colors as an example. In my case, it will be IPFS links. – FrenchieiSverige Jan 4 at 17:32

Fair enough. You will probably want to log and return bytes32 instead of the integers. If this some kind of name service, then you need getters with keys, e.g. function getValue(bytes32 key), and setters with both, e.g. functions(bytes32 key, bytes32 value). You can use contract storage for current state, and you can get your history from events unless there's an important detail I'm missing. LogUpdate(bytes32 key, bytes32 value). I usually default to log all state changes, so there would be LogNew, LogUpdated, LogDeleted, etc. – Rob Hitchens Jan 4 at 17:55

When you are talking about name service, you are referencing some kind of hash tables right? – FrenchieiSverige Jan 5 at 10:19

1 Sorry. Your contract that returns links sounded like name resolution to me. It was just a guess. I've edited the answer to just call it "x" and clarify how you can track the changes. You would be able to reproduce every state x went through with a web3 client. I'm starting to wonder if you mean that you want to access that history inside the contract. If so, "There's a more gas-intensive, storage-intensive approach ..." would apply. Solvable but not trivial. – Rob Hitchens Jan 25 at 12:06

You can call const functions on older states by specifying the blockNumber in the argument list. A good way to find the blockNumber for state changes is by emitting an event. I created an example which shows this here:

https://github.com/chafey/ethereum-events

And a blog entry too:

http://chafey.blogspot.com/2017/03/applying-blockchain-to-healthcare-part_4.html

Here is a link to the code that iterates over the events and gets the older state for the smart contracts:

https://github.com/chafey/ethereum-events/blob/master/app/client/patientChanges.js#L6

answered Mar 5 at 18:45

Chris Hafey
**205** 6