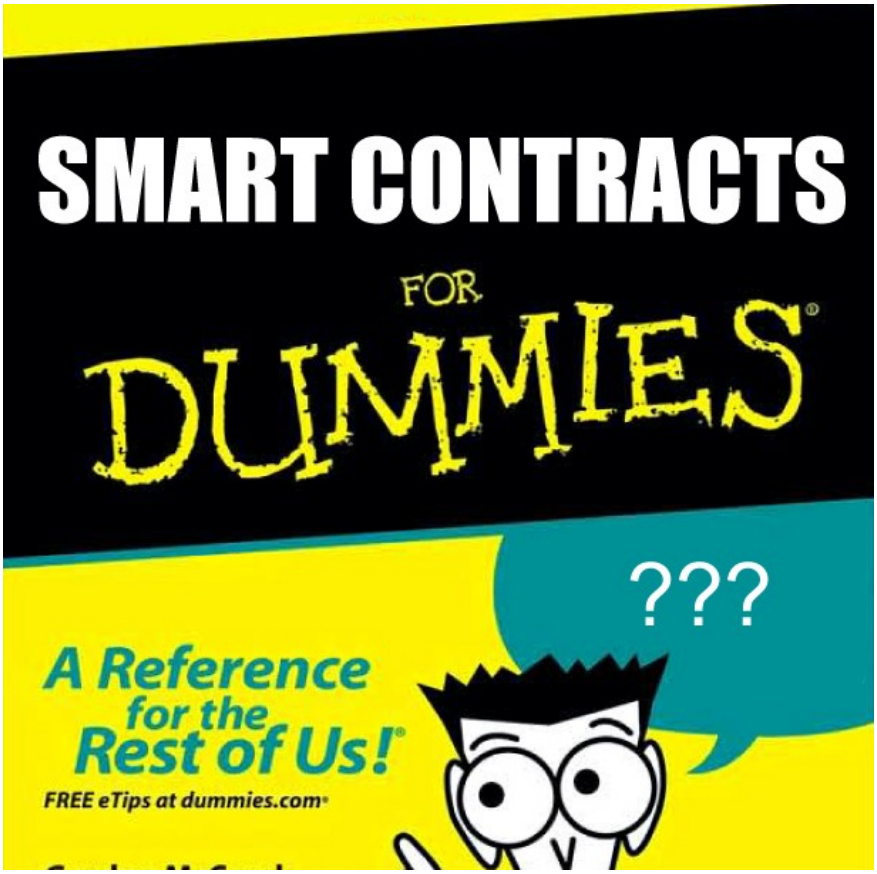
Smart Contracts for Dummies

If you still don't get what the heck a Smart Contract is...

Nik Custodio Follow
May 26, 2017 · 7 min read



Ok, you know a bit about Bitcoin (see: Explain Bitcoin Like I'm Five). You've been seeing the blockchain on the news.

But what's this new **Ethereum** thing? Apparently it's this crypto-currency you can use to build "smart contracts". Sounds impressive. So, uh… what are they again? *(Spoiler: They're not that smart. And they're not really contracts!)*

Instead of a one line definition, let's try to get an intuition. First, we'll revisit the *blockchain* and the word "trust". Then, we'll talk about the

word “contract”. Understanding both words is the secret.

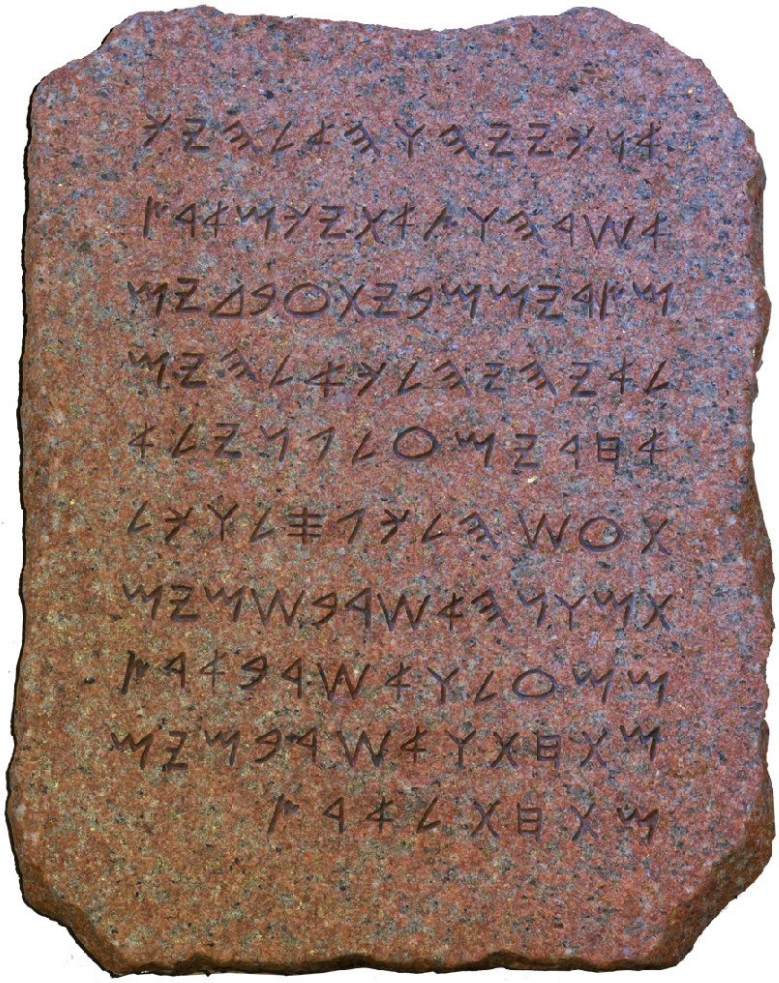
Part I: What we mean by ”Trust(less)”

Most of the time, when we think Bitcoin (or Ethereum), we have a mental image of, well…*coins*.

Aren’t these *crypto-currencies* after all? Isn’t that the whole point? In our minds we see objects—digital gold, or silver (or tulips for the skeptics).

Because these images are easy to understand, we forget a bit about that *thing* that’s underneath it all. So, I say we start thinking about this in a different way.

Digital Stone



Ugh, really? Digital rocks? Actually, rocks are pretty useful.

We have this idiom in the english language that goes something like this: “set it in **stone**.”

> *”I’ve reviewed the contract Bob. Looks good. Let’s set this in stone!”*
>
> *”Don’t get too excited Alice, nothing’s in stone yet.”*
>
> *”This is God. I’ve written my 10 commandments on these two stone tablets. You know. Just in case ya'll start getting any funny ideas.”*

This metaphor continues to have meaning in a modern world because in the physical (ancient) world, stone had some interesting properties:

1. When you carve something on stone there is a physical ***finality and permanence*** to it. You can't make changes just like that.
2. If you try to "erase" something later on, it'll be obvious. Any changes you make to it are quite ***transparent and tamper proof (provable)***.
3. These rules apply equally to all. Stone is ***neutral***. It obeys the laws of physics, not men. It doesn't care if you're a powerful king or a peasant—it behaves exactly the same for everyone.

Because of all these properties, we have a pretty ***high level of trust*** in stone.

I mean—there's a reason why we never say "let's set this agreement in ***sand.***" Stone is the kind of thing I can point to in the future for evidence. Stone equals solid proof—not just any material will do!

The Economist agrees!

When it comes down to it, a blockchain is really just the above: *a kind of material that, through a special mix of cryptography and decentralization, has the properties of permanence, transparency, and neutrality—whatever you put on it.*

Whether it's a list of how many apples you sent to Joe. Or the words "I love Jenny." It doesn't matter. When you put it on a blockchain—it's *on*.

> *Setting something on a blockchain is like setting something in stone. It makes trust easier.*

Except now we can do it *digitally*. And that's pretty special.

Thinking about a blockchain as a piece of stone you can write things on (instead of a piece of currency) also helps us understand its broad potential. Which leads us to…contracts!

. . .

Part II: What we mean by "Contracts"

The word "contract" has a lot of baggage. We start thinking: legal documents and lawyers.

The go-to description used in the news is a bit better: things that *self-execute* or *execute automatically*. That seems vaguely familiar though. After all, there's really nothing new about *automation* or *execution*.

The great grandfather of smart contracts

Take your good ol' office **vending machine** for example. It's a "stupid" machine that does what it's told, and executes things automatically. It's been around for decades!

Behold, the magical machine that spits out high-fructose sustenance.

Let's pretend one afternoon you find yourself in front of this machine. It says: **"If you give me $2.50, and press this button, you will get a Diet Coke."**

It might not actually *say* those words anywhere. But that's the promise of this little interaction. One might even call this a kind of simple ***agreement***. (You can guess where this is going.)

You feed in the money. Press the button. Presto! Bottle in hand, you forget this meaningless event in your life 2 seconds later and start worrying again about those darn TPS reports you forgot to do.

. . .

Well, you didn't notice, but this whole thing was actually a small program("contract") coded("written") into the machine beforehand that ran when you hit the button("signed off on it"). Something like:

```
> if money received == $2.50
>     && the button pressed is "Diet Coke"
> then release Diet__Coke
```

Computer code, as you see, is *kind of* like a ***contract***.

It's making statements and declarations. There are terms (if you do this…then…). And just like someone you trust—it even fulfills its end of the bargain!

Voila. Contracts are just code. But unlike a "contract" in English, this is something both humans *and* machines can read. Extra fun!

Ok, but…

Now you're more confused about this smart contract business. As we said, this is nothing special. In fact, as the vending machine demonstrates, this kind of code *is* already everywhere in our daily lives. If a smart contract is just "if…then" code (or *any* code for that matter), then what's the hoopla? What's actually *new?*

Vending Machines 2.0

One sunny day, you spot a vending machine sitting on the corner. You've never seen this one before!

You walk over and take a look. This machine says: "**If you put in $1,000 this machine will give you $5,000.**"

Whoa! Whoever put this machine together must be very rich and generous.(Or insanely stupid…). Either way. 1k for 5k? No brainer—that's a deal you'll take any day! Right?

This is exactly like our good old Diet Coke machine. Same logic. Same if-then process.

Except now the stakes are different. You reach for your pocket but suddenly, you feel hesitant. Who the hell put this machine together anyway? And what if it eats your money? $1,000 isn't a small amount—you were saving that for months. You didn't think twice about that Diet Coke. But now? Now you realize that maybe vending machines aren't that simple.

You start thinking about **trust.**

> **How do we know it has enough funds to spit out the promised $5,000?**

> **How do we know the code is going to *run*?**

> **Is there any way to publicly and transparently verify this code?**

. . .

Conclusion

The $5,000 vending machine is an extreme, theoretical example but it does hint at the problem with *scaling* trust. In an expanding, digital world where people can connect anonymously—trust becomes a tricky thing. We usually rely on third-parties and other middle men for that reason. We have to. Especially if we're moving things way more valuable than Diet Cokes. You know, like newfangled financial stuff. Or the very idea of "value" and "ownership" itself.

Hmmm. If only you could marry the automation of traditional programming ***and*** the trust-worthy properties of *digital stone*….

Well, that is exactly what a smart contract is! It's just code—with a very special kind of backing.

> *Keep in mind, we've had both computation and execution before. But never one that was finalized in a neutral, provable, trustable way on (digital) stone.*

How about the real world? A few ideas.

Online Gaming: Fight fraud on gambling sites. Are the odds of that dice roll you just did *actually* 1 in 6? How do we know they're going to pay out? Well, why not "set the code in stone" and prove it? A live example.

Supply Chains: Maybe track and verify where and how things are made?

Voting: Maybe a tamper proof voting process?

Decentralized and Autonomous Companies: Sci-fi time.

Throughout history, automation has always been applied to the *bottom* of companies. The assembly line. The factory worker. But if the rules of a corporation are just a kind of operational *logic*— then isn't it possible to flip the pyramid and instead automate the *top*?

. . .

A "Tooth Fairy" smart contract. Coming soon to a sophisticated child near you.

These are only a few examples of what you could code on a blockchain using Ethereum's Turing complete programming language. We're only at the beginning. If you dream it, you might be able to code it.

And in many ways, that's what makes this whole thing exciting. We have some guesses, but honestly—we have no idea what will be built in the years and decades to come.

All we know is that the building blocks are here. And it is open to all. The rest is up to you.

. . .

Ready to go down the rabbit hole?

- A Beginner's Guide to Ethereum
- Crypto-Tokens: A Breakthrough in Open Network Design *(Chris Dixon)*
- The Idea of Smart Contracts *(Nick Szabo, 1997)*
- The Other Side of the Coin: A different perspective on cryptocurrencies