

Like many ideas in the blockchain industry, a general confusion shrouds so called 'smart contracts'.

A new technology made possible by public blockchains, smart contracts are difficult to understand because the term partly confuses the core interaction described.

While a standard contract outlines the terms of a relationship (usually one enforceable by law), a smart contract enforces a relationship with cryptographic code.

Put differently, smart contracts are programs that execute exactly as they are set up to by their creators.

First conceived in 1993, the idea was originally described by computer scientist and cryptographer Nick Szabo as a kind of digital vending machine. In his [famous example](https://en.wikipedia.org/wiki/Smart_contract), he described how users could input data or value, and receive a finite item from a machine, in this case a real-world snack or a soft drink.

In a simple example, ethereum users can send 10 ether to a friend on a certain date using a smart contract (See our guide "[What is Ether?](https://www.coindesk.com/information/what-is-ether/)").

In this case, the user would create a contract, and push the data to that contract so that it could execute the desired command.

Ethereum is a platform that’s built specifically for creating smart contracts.

But these new tools aren’t intended to be used in isolation. It is believed that they can also form the building blocks for 'decentralized applications' (See: "[What is a Dapp?](https://www.coindesk.com/information/what-is-a-decentralized-application-dapp-ethereum)") and even whole decentralized autonomous companies (See: "[What is a DAO?](https://www.coindesk.com/information/what-is-a-dao-ethereum/)')

**How smart contracts work**

It’s worth noting that bitcoin was the first to support basic smart contracts in the sense that the network can transfer value from one person to another. The network of nodes will only validate transactions if certain conditions are met.

But, bitcoin is limited to the currency use case.

By contrast, ethereum replaces bitcoin's more restrictive language (a scripting language of a hundred or so scripts) and replaces it with a language that allows developers to write their own programs.

Ethereum allows developers to program their own smart contracts, or 'autonomous agents', as the [ethereum white paper](https://github.com/ethereum/wiki/wiki/White-Paper) calls them. The language is 'Turing-complete', meaning it supports a broader set of computational instructions.

Smart contracts can:

* Function as 'multi-signature' accounts, so that funds are spent only when a required percentage of people agree
* Manage agreements between users, say, if one buys insurance from the other
* Provide utility to other contracts (similar to how a software library works)
* Store information about an application, such as domain registration information or membership records.

**Strength in numbers**

Extrapolating that last point, smart contracts are likely to need assistance from other smart contracts.

When someone places a simple bet on the temperature on a hot summer day, it might trigger a sequence of contracts under the hood.

One contract would use outside data to determine the weather, and another contract could settle the bet based on the information it received from the first contract when the conditions are met.

Running each contract requires ether transaction fees, which depend on the amount of computational power required.

As explained in our guide "[How Ethereum Works](https://www.coindesk.com/information/how-ethereum-works/)", ethereum runs smart contract code when a user or another contract sends it a message with enough transaction fees.

The Ethereum Virtual Machine then executes smart contracts in 'bytecode', or a series of ones and zeroes that can be read and interpreted by the network.