

# Ethereum Smart-Contracts



elixir



**Alisina Bahadori**  
**@alisinabh**



# DISCLAIMER

This presentation is for informational purposes only and is **not** financial advice. Cryptocurrency and blockchain, especially involving Ethereum smart contracts, carry risks. Individuals should **conduct their own research** and seek professional advice before making any financial decisions. The presenter and organizers do not assume responsibility for actions taken based on this information.

# What is this talk about? 🤔

- A brief introduction to blockchain and smart contracts
- Basics of interacting with smart-contracts.
- Introduction to existing ethereum libraries and ecosystem in Elixir
- What can you build?

# What is Ethereum?

- Ethereum is a decentralized blockchain platform.
- Ether (ETH) is its native currency, used for fees. (Gas) 
- It aims to create a global, trustless and tamper-resistant system. 
- It enables smart contracts: self-executing code with predefined rules.
- Ethereum facilitates decentralized applications (DApps) development.

# What is a Smart Contract?

- A smart contract is a piece of compiled code uploaded in the blockchain.
- They run on blockchain platforms like Ethereum using EVM.
- Eliminate the need for intermediaries.
- Can power a wide range of decentralized applications.

Decentralized Exchanges, Gaming, Autonomous Organizations, Lending/Borrowing etc.

# What do they help with? (A few examples)

## Decentralized Exchanges (DEX)

Unlike a centralized exchange (Yes they call it CEX now) a decentralized exchange does not have control over your funds or accounts. Instead, using smart contracts, you can exchange from a currency to another without the need to ever trusting any third parties (hence trustless).

## In Game Currencies and Assets

Imagine being able to sell (and buy) the cool sword that you won in an epic fight to the free market. Smart contracts (Tokens and NFTs) can be utilized for that.

**CUT THE CRAP**  
**SHOW ME THE CODE...**

**Storage  
(State)**

```
// SPDX-License-Identifier: WTFPL
pragma solidity ^0.8.0;
```

```
contract ElixirCoin {
    mapping(address => uint256) public balances;
```

**Initiation**

```
    constructor() {
        balances[msg.sender] = 10_000;
    }
```

**Functions**

```
    function send(address receiver, uint256 amount) public {
        require(amount <= balances[msg.sender], "Insufficient Balance");
        balances[msg.sender] -= amount;
        balances[receiver] += amount;
    }
}
```



Initiation

```
defmodule ElixirCoin do
  use GenServer

  def start_link(_) do
    GenServer.start_link(__MODULE__, self())
  end

  def init(msg_sender) do
    {:ok, %{balances: %{msg_sender => 10_000}}}
  end
```

Functions

```
  def handle_call({:send, receiver, amount}, {msg_sender, _}, state) do
    if amount <= (state.balances[msg_sender] || nil) do
      state =
        state
        |> update_in([:balances, msg_sender], &(&1 - amount))
        |> update_in([:balances, receiver], &((&1 || 0) + amount))

      {:reply, :ok, state}
    else
      {:reply, {:error, "Insufficient Balance"}, state}
    end
  end
end
```

```
// SPDX-License-Identifier: WTFPL
pragma solidity ^0.8.0;

contract ElixirCoin {
  mapping(address => uint256) public balances;

  constructor() {
    balances[msg.sender] = 10_000;
  }

  function send(address receiver, uint256 amount) public {
    require(amount <= balances[msg.sender], "Insufficient Balance");
    balances[msg.sender] -= amount;
    balances[receiver] += amount;
  }
}
```

Storage  
(State)

# Smart contracts

1. Are immutable (Once deployed cannot change)
2. Store their state on-chain and can only access on-chain data
3. Are predictable (Will give same result in the same state e.g. no randomness)
4. Can be called by anyone!
5. Not smart at all!

## Demo Time (To see is to believe)

May the gods of demo bless this demo with flawless execution and spare us from the dreaded glitches. Let the Wi-Fi be strong, the code be bug-free, and the demo demons be banished.

In the name of the Demo Gods, amen.



Thank you for listening ❤️



**Alisina Bahadori**  
**@alisinabh**