

Contratos Inteligentes

Cláudio Henrique Oliveira Ribeiro
Gabriel Olímpio Rocha Leão
Vinicius Fleury Barbosa



Roteiro

- Definição
- Funcionalidade
- Criando um Smart Contract
 - Na rede Ethereum
 - Na rede Hyperledger
- Iniciando um Smart Contract
- Smart Contract com HyperLedger Fabric



Smart Contract

O que é?

- Um contrato inteligente é um programa de computador que controla direta e automaticamente a transferência de ativos digitais entre as partes sob determinadas condições.
- Funciona da mesma maneira que um contrato tradicional, ao mesmo tempo em que aplica automaticamente o contrato.
- São programas que executam exatamente como são configurados (codificados, programados) por seus criadores.
- Assim como um contrato tradicional é aplicável por lei, os contratos inteligentes são aplicáveis por código.





Funcionalidade

- Um contrato inteligente é apenas um contrato digital com a codificação de segurança do blockchain.
- Ele tem detalhes e permissões escritos em código que exigem que uma sequência exata de eventos ocorra para acionar o acordo dos termos mencionados no contrato inteligente.
- Também pode incluir as restrições de tempo que podem introduzir prazos no contrato.



Lógica Se-Então

A ideia por trás é bastante simples, eles são executados baseados na lógica SE-ENTÃO, por exemplo:

- SE você me enviar o objeto A, ENTÃO a soma (em dinheiro, em criptomoeda) será transferida para você
- SE você transferir uma certa quantidade de ativos digitais (criptomoeda, por exemplo, ether), ENTÃO o objeto A será transferido para você
- SE eu terminar o trabalho, ENTÃO os ativos digitais mencionados no contrato serão transferidos para mim





- Podemos adicionar a restrição QUANDO para incluir o fator tempo nos contratos inteligentes.
- Pode-se ver que esses contratos inteligentes ajudam a definir as condições em que devem ser cumpridas para que os termos do contrato sejam executados.
- Não há limite de quanto SE ou ENTÃO você pode incluir em seu contrato inteligente.



Criando um Smart Contract



Na rede Ethereum

Para fazer a criação do Contrato Inteligente, o programador precisa ter acesso a uma interface de Deploy de contratos, e para fazer esse Deploy precisamos de uma quantidade em Ether que é utilizada como recompensa ao minerador responsável por criar esse contrato.



Na rede Hyperledger

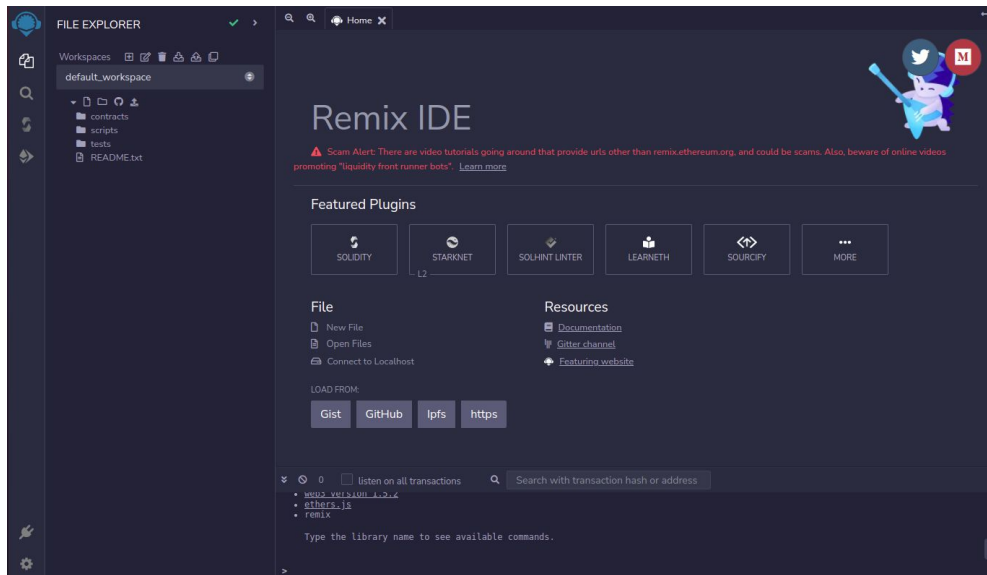
- Diferentemente do Bitcoin, os contratos inteligentes nesse sistema são invocados por um software chamado Chaincode. Esses contratos inteligentes normalmente só interagem com o estado global e são implementados em linguagens como Go, Java e etc.
- Um **chaincode** normalmente é usado pelos administradores para agrupar contratos inteligentes relacionados à confirmação, mas também pode ser usado para a programação de sistema de baixo nível da Fabric.



Iniciando um Smart Contract

Remix Online Editor

- Remix Online Editor é uma IDE usada para a implementação de smart contracts.



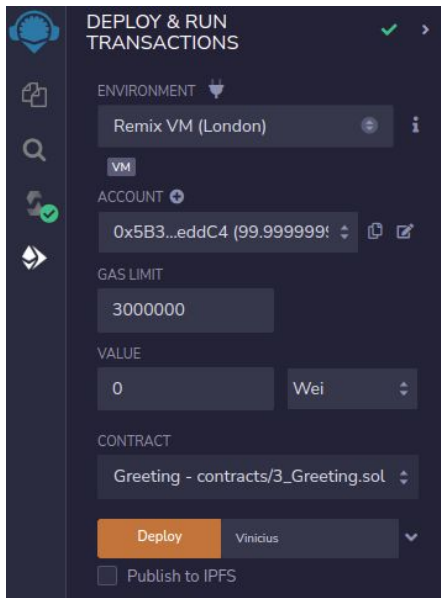
Greeting.

```
1 //SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.11;
3
4 contract Greeting {
5     string public name;
6     string public gp = "Hello ";
7
8     constructor(string memory initialName) {
9         name = initialName;
10    }
11
12    function setName(string memory newName) public {
13        name = newName;
14    }
15
16    function getGreeting() public view returns (string memory) {
17        return string(abi.encodePacked(gp, name));
18    }
19 }
```

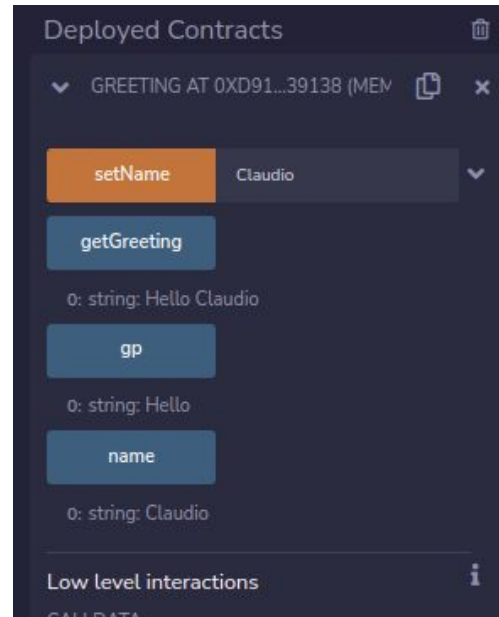
Esse código é um exemplo de smart contract básico, ele realiza a concatenação de um nome com a mensagem Hello.

É importante destacar que para a realização de um smart contract é preciso de uma licença, no caso do código MIT.





Depois do código compilado, vemos suas propriedades, para o início utilizamos uma máquina virtual, e algumas propriedades padrões. Selecionamos o Deploy, que insere o smart contract na blockchain.



Aqui podemos ver as funções e variáveis do nosso contrato. Caso você queira mudar o nome é só setar um nome diferente que a função retorna uma string com o nome escolhido.





Smart Contract com HyperLedger Fabric



HyperLedger Fabric

```
Txn = issue  
Issuer = MagnetoCorp  
Paper = 00001  
Issue time = 31 May 2020 09:00:00 EST  
Maturity date = 30 November 2020  
Face value = 5M USD
```

```
async issue (ctx, issuer, paperNumber, issueDateTime, maturityDateTime, faceValue) {...}
```

```
Txn = buy  
Issuer = MagnetoCorp  
Paper = 00001  
Current owner = MagnetoCorp  
New owner = DigiBank  
Purchase time = 31 May 2020 10:00:00 EST  
Price = 4.94M USD
```

```
async buy(ctx, issuer, paperNumber, currentOwner, newOwner, price, purchaseTime) {...}
```



HyperLedger Fabric

[\[GitHub\] Código fonte contrato](#)

```
async issue(ctx, issuer, paperNumber, issueDateTime, maturityDateTime, faceValue) {  
    // create an instance of the paper  
    let paper = CommercialPaper.createInstance(issuer, paperNumber, issueDateTime, maturityDateTime, faceValue);  
  
    // Smart contract, rather than paper, moves paper into ISSUED state  
    paper.setIssued();  
  
    // Newly issued paper is owned by the issuer  
    paper.setOwner(issuer);  
  
    // Add the paper to the list of all similar commercial papers in the ledger world state  
    await ctx.paperList.addPaper(paper);  
  
    // Must return a serialized paper to caller of smart contract  
    return paper.toBuffer();  
}
```



HyperLedger Fabric

```
async buy(ctx, issuer, paperNumber, currentOwner, newOwner, price, purchaseDateTime) {  
  // Retrieve the current paper using key fields provided  
  let paperKey = CommercialPaper.makeKey([issuer, paperNumber]);  
  let paper = await ctx.paperList.getPaper(paperKey);  
  
  // Validate current owner  
  if (paper.getOwner() !== currentOwner) {  
    throw new Error('Paper ' + issuer + paperNumber + ' is not owned by ' + currentOwner);  
  }  
  
  // First buy moves state from ISSUED to TRADING  
  if (paper.isIssued()) {  
    paper.setTrading();  
  }  
  
  // Check paper is not already REDEEMED  
  if (paper.isTrading()) {  
    paper.setOwner(newOwner);  
  } else {  
    throw new Error('Paper ' + issuer + paperNumber + ' is not trading. Current state = ' + paper.getCurrentState());  
  }  
  
  // Update the paper  
  await ctx.paperList.updatePaper(paper);  
  return paper.toBuffer();  
}
```



Obrigado!



Referências

- [O que são smart contracts?](#)
- [Smart Contracts: entenda o que são e como funcionam](#)
- [O que é Ethereum?](#)
- [Blockchain - Hyperledger vs Ethereum](#)
- [\[Video\] Smart Contracts 101](#)
- [Writing Your First Application](#)
- [Blockchain | Smart Contracts - GeeksforGeeks](#)
- [Contratos Inteligentes e Chaincode – Documentação](#)
- [Processamento de Contrato Inteligente](#)
- [\[GitHub\] Código fonte contrato de exemplo](#)
- [Contratos Inteligentes e Chaincode](#)
-

