区块链开发入门

Solidity智能合约开发
从入门到精通

第1课：智能合约基础与环境搭建

讲师：区块链导师Layer

今天的学习内容

1. 什么是智能合约?

深入理解智能合约的定义、核心特性及其与传统合约的区别

2. 以太坊虚拟机（EVM）基础

探索EVM作为智能合约运行环境的关键作用和基本工作机制

3. 开发环境搭建 – Remix IDE

学习如何使用Remix IDE这一便捷的在线开发工具，无需本地配置即可开始智能合约开发

4. 编写第一个智能合约

亲自动手编写一个简单的Solidity智能合约，了解其基本结构

5. 编译和部署实战

掌握在Remix IDE中编译和部署智能合约到测试环境的实际操作步骤

6. 总结与作业

对本课内容进行回顾总结，并布置课后作业以巩固所学知识

课程目标

本次课程旨在为初学者提供Solidity智能合约开发的核心概念理解
和基础开发环境搭建指导，为后续深入学习奠定坚实基础

适合零基础学习

智能合约 Smart Contract

定义

运行在区块链上的程序，以数字形式规定了承诺，并能自动执行这些承诺的协议

传统程序

- 代码可以被修改或删除
- 执行过程不透明，无法验证
- 需要第三方干预或人工操作

智能合约

- 一旦部署，代码永久存在，无法修改
- 所有交易记录和执行过程公开可查
- 满足条件时自动执行，无需人工干预

"智能合约就像刻在石头上的法律，一旦立下，便无法更改"

智能合约的三大特点

不可篡改

一旦智能合约部署到区块链上，其代码和数据就永久存在，无法被修改或删除。

确保了合约执行的确定性和可靠性，如同刻在石头上的法律

公开透明

智能合约的所有交易记录和执行过程都在区块链上公开可查，任何人都可以验证其逻辑和结果。

这种透明性消除了信息不对称，增强了信任

自动执行

当预设的条件被满足时，智能合约会自动运行，无需任何第三方干预或人工操作。

大大提高了效率，降低了执行成本，并避免了人为错误和舞弊

这些特点共同构成了智能合约的核心优势，为区块链技术提供了坚实基础

实际应用案例：智能合约打赌

对比传统打赌方式与智能合约方式，看看智能合约如何改变游戏规则

传统方式

参与方：
你 ⟷ 朋友

信任机制：
需要互相信任

履约风险：
可能赖账

执行方式：
需要第三方担保

效率与成本：
低效率，高成本

VS

智能合约方式

参与方：
你 → 智能合约 ← 朋友

信任机制：
不需要信任

履约风险：
不可能赖账

执行方式：
完全自动化
通过预言机获取天气数据，自动转账给赢家

效率与成本：
高效率，低成本

智能合约能做什么？

数字货币

USDT、USDC等稳定币的基础，确保数字资产的发行、流通和赎回规则自动化执行

NFT艺术品

非同质化代币的创建、交易和所有权管理，实现数字艺术品和收藏品的独特性和可追溯性

去中心化金融

DeFi协议（借贷、交易、流动性挖矿）的核心逻辑，构建无需传统金融中介的开放式金融系统

供应链溯源

记录商品从生产到消费的每一个环节，实现商品流转的透明追踪，有效防止假冒伪劣

投票系统

构建透明、公正的投票系统，确保投票过程的不可篡改性和结果的公开性

游戏道具

在区块链游戏中，虚拟物品（如游戏道具、角色）的所有权和交易通过智能合约管理，赋予玩家真正的数字资产所有权

可能性无限！

以太坊虚拟机 EVM

什么是EVM?

以太坊虚拟机（Ethereum Virtual Machine）可以被理解为一个全球性的、去中心化的超级计算机，它运行在以太坊网络的成千上万个节点上。

核心职责

EVM的核心职责是执行智能合约，通过提供一个隔离的、确定性的运行时环境，确保在以太坊上部署的代码在任何地方都能以相同的方式运行，并产生完全相同的结果。

保障机制

这种机制保障了智能合约的去中心化和安全性，无需任何中央权威机构的干预，即可实现可靠的、不可篡改的执行环境。

从代码到部署

流程说明

1. 智能合约开发人员编写Solidity代码，定义合约的逻辑和功能
2. 使用Solidity编译器将代码编译成EVM可执行的字节码
3. 生成的字节码包含合约的所有功能和状态变量
4. 通过Remix等工具将合约部署到以太坊区块链测试网络
5. 部署成功后，全球节点开始运行该合约，任何人都可以与其交互，前提是合约有公共函数暴露出来

重要概念：Gas费用

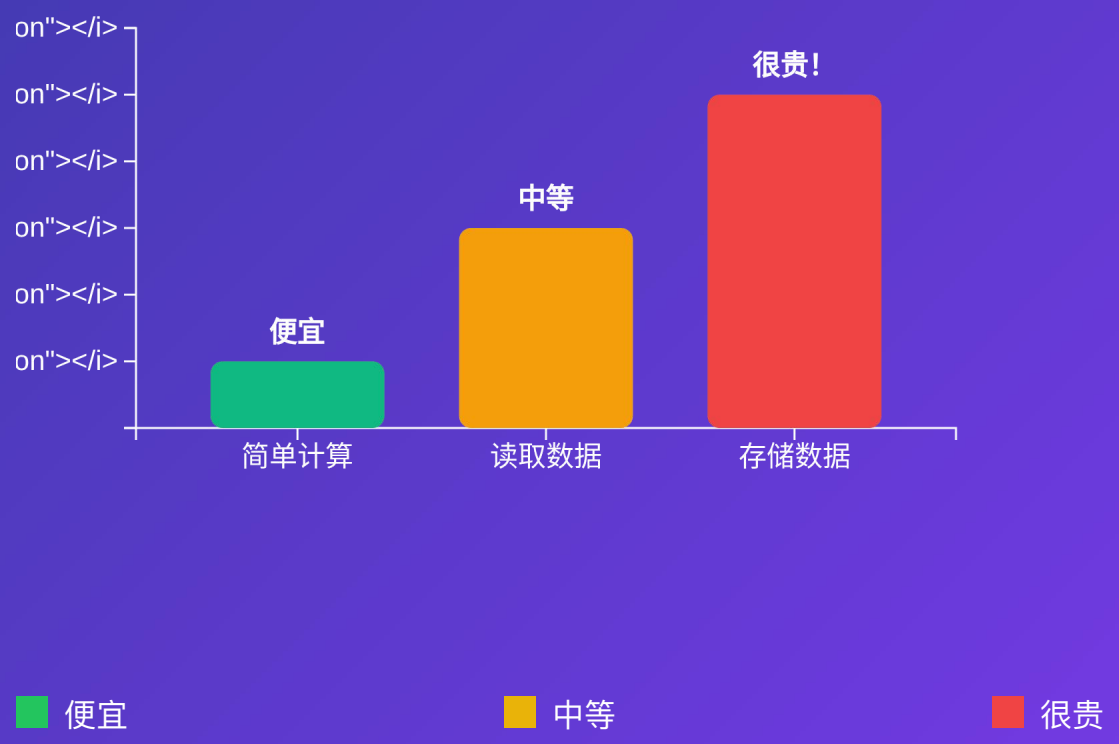
什么是Gas?

Gas是以太坊网络上执行操作时所需的计算工作量的计量单位，是理解智能合约运作机制的核心概念。

它类似于汽车所需的燃油，执行智能合约需要支付相应的Gas费用。Gas的主要目的在于保护网络安全和合理分配资源，防止恶意用户通过运行无限循环的计算来瘫痪整个网络。

优秀开发者的目标是编写省Gas的代码，以优化合约的执行效率和降低成本。

操作Gas成本对比



开发工具：Remix IDE

Remix IDE

智能合约开发的瑞士军刀

remix.ethereum.org

无需安装

直接在浏览器中运行，无需安装Solidity编译器或节点环境，开箱即用

功能完整

集成了编译、部署、调试等全套功能，简化了复杂流程

官方维护

由以太坊官方维护，稳定可靠，确保与最新Solidity版本和EVM兼容

适合学习

直观的图形界面和内置调试器，是学习Solidity智能合约的最佳入门工具

Remix IDE是智能合约开发的首选工具，无需配置即可开始Solidity开发之旅

Remix界面介绍

1

文件管理

左侧面板，管理智能合约文件和项目结构

2

代码编辑

中间区域，编写和编辑Solidity智能合约代码

3

功能面板

右侧面板，提供编译、部署、测试等操作功能

4

控制台

底部区域，显示编译结果、部署信息和交互日志

通过拖拽调整各区域大小，创建新文件，保存项目到GitHub等

智能合约基本结构


```
// SPDX-License-Identifier: MIT ← 许可证声明
pragma solidity ^0.8.0; ← 版本声明

contract HelloWorld { ← 合约定义

    string public message; ← 状态变量

    constructor() { ← 构造函数
        message = "Hello, Solidity!";
    }

    function updateMessage(string memory _newMessage) public { ← 公共函数
        message = _newMessage;
    }

    function getMessage() public view returns (string memory) { ← 视图函数
        return message;
    }
}
```

许可证声明

指定代码遵循的开源许可证，如MIT。

版本声明

指定兼容的Solidity编译器版本，^0.8.0表示0.8.0到0.9.0（不含）。

合约定义

使用 Contract

重要：Solidity版本

版本声明示例

```
pragma solidity ^0.8.0;
```

^符号表示什么？

在Solidity中，版本号前的^符号表示：

^0.8.0 = 0.8.0 到 0.9.0（不含）

允许的版本

- 0.8.0
- 0.8.1
- 0.8.19
- 0.8.20

不允许的版本

- 0.9.0
- 0.7.x

为什么指定版本很重要？

- 不同版本语法可能不同
- 确保代码兼容性

编译和部署流程

在Remix IDE中，智能合约的编译和部署是一个直观的四步流程：

1 编译

Solidity Compiler
将Solidity代码编译为EVM字节码

- 点击左侧"Solidity Compiler"图标
- 选择合适的编译器版本
- 点击"Compile"按钮

编译成功后会显示绿色对勾

2 选择环境

Deploy & Run Transactions
选择运行环境

- 切换到"Deploy & Run Transactions"插件
- 在"Environment"下拉菜单中选择
- 选择Remix VM（测试用）

不消耗真实以太币

3 部署合约

Deploy Contract
将合约部署到区块链

- 确保选择要部署的合约
- 如有构造函数参数，请填入

4 测试功能

测试合约功能
与部署的合约交互

- 在"Deployed Contracts"区域展开合约
- 点击公共函数按钮

恭喜你！

你已经成功：

- 编写了第一个智能合约
- 编译通过
- 部署到区块链
- 调用了合约函数

现在你是智能合约开发者了！

继续学习，探索更多智能合约的奇妙世界！

今日总结

今天我们学习了智能合约开发的基础知识，涵盖了以下五个核心要点：

智能合约的三大特点

不可篡改、公开透明和自动执行，这些特点使其在区块链环境中具有独特的优势

EVM是运行环境

以太坊虚拟机（EVM）作为全球共享的去中心化计算机，是智能合约得以安全、确定性运行的基础

Gas费用很重要

执行智能合约需要消耗Gas，理解Gas机制并优化代码以节省Gas成本是成为优秀开发者的关键

Remix是最佳学习工具

因其无需安装、功能完整、官方维护和适合学习的特点，成为智能合约初学者的理想选择

智能合约基本结构

包括许可证声明、版本声明、合约定义、状态变量、构造函数和公共函数

今日作业

任务：扩展HelloWorld合约

在本作业中，你需要扩展我们课堂上编写的HelloWorld智能合约， 添加一个记录合约创建者的功能。

要求

1. **添加状态变量 owner**
   类型：address，记录合约创建者
2. **添加函数 getOwner()**
   返回创建者地址
3. **部署并测试**
   将修改后的合约部署到Remix并测试其功能

代码示例

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

contract HelloWorld {

    string public message;
    address public owner; // 添加状态变量

    constructor() {
        message = "Hello, Solidity!";
        owner = msg.sender; // 在构造函数中初始化
    }

    function updateMessage(string memory _newMessage) public {
        message = _newMessage;
    }

    function getMessage() public view returns (string memory) {
        return message;
    }

    function getOwner() public view returns (address) { // 添加函数
        return owner;
    }
}
```

下节课预告

第2课：EVM存储结构

三种存储方式

storage、memory、calldata 三种存储方式的区别与应用场景

Gas成本分析

为什么 storage 如此昂贵？深入理解 Gas 成本机制

存储优化技巧

如何优化存储，掌握节省 Gas 费用的专业技巧

敬请期待！

感谢观看！

我们已经完成了智能合约开发的基础课程

下次课再见！