Smart Contract Overview

Welcome to the Smart Contract Getting Started guide. This overview explains the basic concepts of smart contract development and oracle networks.

Skip ahead:To get your hands on the code right away, you can skip this overview:

- Deploy Your First Smart Contract : If you are new to smart contracts, deploy your first smart contract in an interactive web development environment.
- Learn how to use Data Feeds : If you are already familiar with smart contracts and want to learn how to createhybridsmart contracts, use Chainlink Data Feeds to get asset price data onchain.

What is a smart contract? What is a hybrid smart contract?

When deployed to a blockchain, asmart contractis a set of instructions that can be executed without intervention from third parties. The smart contract code defines how it responds to input, just like the code of any other computer program.

A valuable feature of smart contracts is that they can store and manage onchain assets (likeETH or ERC20 tokens), just like you can with an Ethereum wallet. Because they have an onchain address like a wallet, they can do everything any other address can. This enables you to program automated actions when receiving and transferring assets.

Smart contracts can connect to real-world market prices of assets to produce powerful applications. Securely connecting smart contracts with offchain data and services is what makes themhybridsmart contracts. This is done using oracles.

What language is a smart contract written in?

The most popular language for writing smart contracts on Ethereum and EVM Chains isSolidity . It was created by the Ethereum Foundation specifically for smart contract development and is constantly being updated. Other languages exist for writing smart contracts on Ethereum and EVM Chains, but Solidity is the language used for Chainlink smart contracts.

If you've ever written Javascript, Java, or other object-oriented scripting languages, Solidity should be easy to understand. Similar to object-oriented languages, Solidity is considered to be acontract-oriented language.

Some networks are not EVM-compatible and use languages other than Solidity for smart contracts:

- Solana * Writing Solana contracts in Rust
- Writing Solana contracts in C

What does a smart contract look like?

The structure of a smart contract is similar to that of a class in Javascript, with a few differences. For example, the followingHelloWorldcontract is a simple smart contract that stores a single variable and includes a function to update the value of that variable.

// SPDX-License-Identifier: MITpragmasolidity0.8.7;/* * *THIS IS AN EXAMPLE CONTRACT THAT USES UN-AUDITED CODE. * DO NOT USE THIS CODE IN PRODUCTION.* /contractHelloWorld{stringpublicmessage;constructor(stringmemoryinitialMessage) {message=initialMessage;}functionupdateMessage(stringmemorynewMessage)public{message=newMessage;}} Open in Remix What is Remix?

Solidity versions

The first thing that every Solidity file must have is the Solidity version definition. TheHelloWorld.solcontract uses version0.8.7, which is defined in the contract aspragma solidity 0.8.7;

You can see the latest versions of the Solidity compilerhere . You might also notice smart contracts that are compatible with a range of versions.

pragmasolidity>=0.7.0<0.9.0; This means that the code is written for Solidity version 0.7.0, or a newer version of the language up to, but not including version 0.9.0. Thepragmaselects the compiler, which defines how the code is treated.

Naming a Contract

Thecontractkeyword defines the name of the contract, which in this example isHelloWorld. This is similar to declaring aclassin Javascript. The implementation ofHelloWorldis inside this definition and denoted with curly braces.

contractHelloWorld{}

Variables

Like Javascript, contracts can have state variables and local variables.State variablesare variables with values that are permanently stored in contract storage. The values oflocal variables, however, are present only until the function is executing. There are also different types of variables you can use within Solidity, such asstring,uint256, etc. Check out theSolidity documentation to learn more about the different kinds of variables and types.

Visibility modifiersare used to define the level of access to these variables. Here are some examples of state variables with different visibility modifiers:

stringpublicmessage;uint256internalinternalVar;uint8privateprivateVar; Learn more about state variables visibilityhere .

Constructors

Another familiar concept to programmers is theconstructor. When you deploy a contract, the constructor sets the state of the contract when it is first created.

InHelloWorld, the constructor takes in astringas a parameter and sets themessagestate variable to that string.

constructor(stringmemoryinitialMessage){message=initialMessage;}

Functions

Functionscan access and modify the state of the contract or call other functions on external contracts.HelloWorldhas a function namedupdateMessage, which updates the current message stored in the state.

constructor(stringmemoryinitialMessage)
{message=initialMessage;}functionupdateMessage(stringmemorynewMessage)public{message=newMessage;} Functions use visibility modifiers to define the access level. Learn more about functions visibilityhere .

Interfaces

Aninterfaceis another concept that is familiar to programmers of other languages. Interfaces define functions without their implementation, which leaves inheriting contracts to define the actual implementation themselves. This makes it easier to know what functions to call in a contract. Here's an example of an interface:

// SPDX-License-Identifier: MITpragmasolidity0.8.7;/* * *THIS IS AN EXAMPLE CONTRACT THAT USES UN-AUDITED CODE. * DO NOT USE THIS CODE IN PRODUCTION.*
/interfacenumberComparison{functionisSameNum(uinta,uintb)externalviewreturns(bool);}contractTestisnumberComparison{constructor()
{}functionisSameNum(uinta,uintb)externalpureoverridereturns(bool){if(a==b){returntrue;}else{returnfalse;}}} Open in Remix What is Remix? For this example,overrideis necessary in theTestcontract function because it overrides the base function contained in thenumberComparisoninterface. The contract usespureinstead ofviewbecause theisSameNumfunction in theTestcontract does not return a storage variable.

What does "deploying" mean?

Deployinga smart contract is the process of pushing the code to the blockchain, at which point it resides with an onchain address. Once it's deployed, the code cannot be changed and is said to beimmutable.

As long as the address is known, its functions can be called through an interface, orEtherscan , or through a library likeweb3js ,web3py ,ethers , and more. Contracts can also be written to interact with other contracts on the blockchain.

What is a LINK token?

The LINK token is an ERC677 token that inherits functionality from theERC20 token standard and allows token transfers to contain a data payload. It is used to pay node operators for retrieving data for smart contracts and also for deposits placed by node operators as required by contract creators.

Any wallet that handles ERC20 tokens can store LINK tokens. The ERC677 token standard that the LINK token implements still retains all functionality of ERC20 tokens.

What are oracles?

Oraclesprovide a bridge between the real-world and onchain smart contracts by being a source of data that smart contracts can rely on, and act upon.

Oracles play a critical role in facilitating the full potential of smart contract utility. Without a reliable connection to real-world conditions, smart contracts cannot effectively serve the real-world.

How do smart contracts use oracles?

Oracles are most popularly used withData Feeds . DeFi platforms likeAAVE andSynthetix use Chainlink data feed oracles to obtain accurate real-time asset prices in their smart contracts.

Chainlink data feeds are sources of dataaggregated from many independent Chainlink node operators . Each data feed has an onchain address and functions that enable contracts to read from that address. For example, theETH / USD feed .

Smart contracts also use oracles to get other capabilities onchain:

- Generate Verifiable Random Numbers (VRF) : Use Chainlink VRF to consume randomness in your smart contracts.

- Call External APIs (Any API) : Request & Receive data from any API using the Chainlink contract library.
- Automate Smart Contracts using Chainlink Automation : Automating smart contract functions and regular contract maintenance.

What is Remix?

Remix is a web IDE (integrated development environment) for creating, running, and debugging smart contracts in the browser. It is developed and maintained by the Ethereum foundation. Remix allows Solidity developers to write smart contracts without a development machine since everything required is included in the web interface. It allows for a simplified method of interacting with deployed contracts, without the need for a command line interface. Remix also has support for samples. This means that Remix can load code from Github.

To learn how to use Remix, see theDeploying Your First Smart Contract guide.

Deploy Your First Smart Contract

What is MetaMask?

Contracts are deployed by other addresses on the network. To deploy a smart contract, you need an address. Not only that, but you need an address which you can easily use with Remix. Fortunately,MetaMask is just what is needed.MetaMaskallows anyone to create an address, store funds, and interact with Ethereum compatible blockchains from a browser extension.