The new Keyword
Creating New Contracts
Step by Step Guide
Exercise
Contract to Contract Interactions
Intro to Interfaces
Calling Another Contract
Testing the Interface
Step by Step Guide
Events
Step by Step Guide
Address and Payable
Guide

Development with Foundry

Deploying a smart contract

Address and Payable

Address and Payable in Solidity

Copy page

A comprehensive guide to understanding and using address and payable address types in Solidity.

Understanding address and payable address types is crucial for managing Ether transfers and interactions within your Solidity contracts. This article will delve into their key distinctions and practical applications.

Objectives

By the end of this lesson, you should be able to:

- Differentiate between address and address payable types in Solidity
- Determine when to use each type appropriately in contract development
- Employ address p able functions

Ask a question... Ctrl+I

Creating New Contracts

Step by Step Guide

Exercise

Intro to Interfaces

Calling Another Contract

Testing the Interface

Step by Step Guide

Step by Step Guide

Guide

Development with Foundry

Deploying a smart contract

Ethereum Addresses

In Solidity, Ethereum addresses play a crucial role in interacting with the Ethereum blockchain. An Ethereum address is a 20-byte hexadecimal string that represents the destination of transactions or the owner of a smart contract. These addresses are used to send and receive Ether and interact with smart contracts.

Addresses

Regular addresses in Solidity are used for various purposes, including:

- Identifying the owner of a smart contract
- Sending Ether from one address to another
- Checking the balance of an address Here’s an example of declaring a regular address variable in Solidity:

```
address public owner;
```

Payable Addresses

`payable` keyword is a language-level feature provided by Solidity to enable the handling of Ether within smart contracts, and it is not a feature of the Ethereum Virtual Machine itself, but rather a part of the Solidity language's syntax. They are used when you want a contract to be able to receive Ether from external sources, such as other contracts or user accounts.

Payable addresses are often used when creating crowdfunding or token sale contracts, where users send Ether to the contract's address in exchange for tokens or to fund a project.

Here's an example of declaring a payable address variable in Solidity:

```
address payable public projectWallet;
```

Payable **Address** are marked as payable, which means they can accept incoming Ether transactions. It's important to note that regular addresses cannot receive Ether directly.

Receiving Ether with Payable Addresses

To receive Ether in a contract using a payable address, you need to define a payable function that can accept incoming transactions. This function is typically named receive or fallback. Here's an example:

```
fallback() external payable {
    // Handle the incoming Ether here
}
```

Creating New Contracts
Step by Step Guide
Exercise
Intro to Interfaces
Calling Another Contract
Testing the Interface
Step by Step Guide
Step by Step Guide
Guide
Development with Foundry
Deploying a smart contract

In this example, the fallback function is marked as external and payable, which means it can receive Ether when someone sends it to the contract’s address. You can then add custom logic to handle the received Ether, such as updating contract balances or triggering specific actions.

Usage

Creating New Contracts
Step by Step Guide
Exercise
Intro to Interfaces
Calling Another Contract
Testing the Interface
Step by Step Guide
Step by Step Guide
Guide
Development with Foundry
Deploying a smart contract

```
contract PaymentReceiver {
    address payable owner;

    constructor() payable {
        owner = payable(msg.sender); // Convert msg.sender to payable
    }

    function receiveEther() public payable {
        // This function can receive Ether
    }

    function withdrawEther() public {
        owner.transfer(address(this).balance); // Send Ether to owner
    }
}
```

Conclusion

Appropriately using address and address payable types is essential for secure and efficient Solidity contract development. By understanding their distinctions and applying them correctly, you can effectively manage Ether transfers and interactions within your contracts.

Was this page helpful? Yes No Suggest edits Raise issue

Step by Step Guide

Deploying a smart contract using Foundry

Creating New Contracts

Step by Step Guide

Exercise

Intro to Interfaces

Calling Another Contract

Testing the Interface

Step by Step Guide

Step by Step Guide

Guide

Development with Foundry

Deploying a smart contract

base docs