参考 [OpenZeppelin文档](https://docs.openzeppelin.com/contracts/4.x/api/utils" \l "IERC165) 并结合自己的理解。

## 什么是ERC165

一种用来检测合约所继承接口的标准方法。[EIP-165](https://github.com/ethereum/EIPs/blob/master/EIPS/eip-165.md) 中提出。

## 为什么要遵守ERC165

EIP-165 中的动机：

For some "standard interfaces" like [the ERC-20 token interface](https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md), it is sometimes useful to query whether a contract supports the interface and if yes, which version of the interface, in order to adapt the way in which the contract is to be interacted with. Specifically for ERC-20, a version identifier has already been proposed. This proposal standardizes the concept of interfaces and standardizes the identification (naming) of interfaces.

继承了 ERC165 的合约方便他人快速检测其实现了哪些接口。

## 代码实现

需实现以下函数：

function supportsInterface(interfaceId)

### IERC165

import "@openzeppelin/contracts/utils/introspection/IERC165.sol";

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

interface IERC165 {

function supportsInterface(bytes4 interfaceId) external view returns (bool);

}

如果合约实现了 interfaceId 定义的接口，则返回 true

### ERC165

import "@openzeppelin/contracts/utils/introspection/ERC165.sol";

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

import "./IERC165.sol";

abstract contract ERC165 is IERC165 {

function supportsInterface(bytes4 interfaceId) public view virtual override returns (bool) {

return interfaceId == type(IERC165).interfaceId;

}

}

#### 补充

表达式 type(x) 可以检测参数 x 的类型信息。

接口类型的特别属性：

type(I).interfaceId :

返回接口 I 的 bytes4 类型的接口 ID，接口 ID 被定义为 XOR (异或) 接口内所有函数的函数选择器(function selector)。

## 实际使用

ERC721合约中对ERC165的重写：

function supportsInterface(bytes4 interfaceId) public view virtual override(ERC165, IERC165) returns (bool) {

return

interfaceId == type(IERC721).interfaceId ||

interfaceId == type(IERC721Metadata).interfaceId ||

super.supportsInterface(interfaceId);

}