Interfaces are a concept in programming languages that separate the declaration of a function from its actual behavior.

In Solidity, interfaces act as contracts or agreements between the interface and any contract that implements it. By using the interface of a contract, users are bound to use the functions declared in the interface within their contract.

The interface serves as the skeleton of a smart contract, providing a rough idea for other smart contracts to build functionalities and implement them within their contracts.

Why do we use Interfaces?

The interface is a tool used to interact with existing smart contracts on the blockchain, such as the ERC20 token standard contract.

Openzeppelin offers an ERC20 interface for customization, allowing users to create ERC20 tokens according to their specifications.

By creating an interface, users can upgrade contracts based on their features and functionalities, but must customize the skeleton and use the declared functions.

For instance, if users want to inherit functions from another contract without access to its code, they can use an interface to call other contracts.

Interfaces Restrictions

So as I mentioned in the introduction, interfaces in solidity are a more restricted form of abstract contracts. Like in abstract contracts we must have at least one function without its implementation, interface adds some more restrictions to that.

Following are the constraints for the interface to use :

1. You can not declare state variables in the interface.
2. You can not use the constructor inside and interface.
3. You are also not allowed to write modifiers in the interface.
4. You can only declare the functions inside the interface and can not define them inside the interface.
5. All declared functions inside the interface must be external.

YouTube Link:

https://www.youtube.com/embed/V1-tLzM_q3c

Solidity Source File