

# FHERC20

## FHERC20

### **\_encBalances**

mapping(address => uint32) \_encBalances

### **constructor**

constructor(string name, string symbol) public

### **\_allowanceEncrypted**

function \_allowanceEncrypted(address owner, address spender) public view virtual returns (uint32)

### **allowanceEncrypted**

function allowanceEncrypted(address spender, struct Permission permission) public view virtual returns (bytes) \_Returns the remaining number of tokens that spender will be allowed to spend on behalf of owner through [transferFromEncrypted](#) . This is zero by default.

This value changes when [approveEncrypted](#) or [transferFromEncrypted](#) are called.\_

### **approveEncrypted**

function approveEncrypted(address spender, struct inEuint32 value) public virtual returns (bool) \_Sets a value amount of tokens as the allowance of spender over the caller's tokens.

Returns a boolean value indicating whether the operation succeeded.

IMPORTANT: Beware that changing an allowance with this method brings the risk that someone may use both the old and the new allowance by unfortunate transaction ordering. One possible solution to mitigate this race condition is to first reduce the spender's allowance to 0 and set the desired value afterwards:

<https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729>

Emits an ApprovalEncrypted event.\_

### **\_approve**

function \_approve(address owner, address spender, uint32 value) internal

### **\_spendAllowance**

function \_spendAllowance(address owner, address spender, uint32 value) internal virtual returns (uint32)

### **transferFromEncrypted**

function transferFromEncrypted(address from, address to, uint32 value) public virtual returns (uint32)

### **transferFromEncrypted**

function transferFromEncrypted(address from, address to, struct inEuint32 value) public virtual returns (uint32) \_Moves a value amount of tokens from from to to using the allowance mechanism. value is then deducted from the caller's allowance.

Returns a boolean value indicating whether the operation succeeded.

Emits a TransferEncrypted event.\_

### **wrap**

function wrap(uint32 amount) public

### **unwrap**

function unwrap(uint32 amount) public

## **\_mintEncrypted**

function \_mintEncrypted(address to, struct inEuint32 encryptedAmount) internal

## **transferEncrypted**

function transferEncrypted(address to, struct inEuint32 encryptedAmount) public returns (euint32)

## **transferEncrypted**

function transferEncrypted(address to, euint32 amount) public returns (euint32)

## **\_transferImpl**

function \_transferImpl(address from, address to, euint32 amount) internal returns (euint32)

## **balanceOfEncrypted**

function balanceOfEncrypted(address account, struct Permission auth) public view virtual returns (bytes) Returns the value of tokens owned byaccount , sealed and encrypted for the caller. [Edit this page](#)

[Previous](#) [Next](#) [Privacy Policy](#)