**Contract Name: DeedzToken**The LockableToken contract manages token locking functionality and extends the ERC1132 and ERC20 contracts. It provides methods to lock tokens, check locked token balances, extend lock duration, increase locked token amount, check unlockable token balances, unlock tokens, and retrieve the total unlockable tokens for an address.

**Contract Inheritance:**ERC1132: Provides the ERC-1132 token locking interface.  
ERC20: Implements the ERC-20 token standard.  
Owned: Abstract contract for managing ownership functionality.

**Dependencies:**SafeMath: A library for safe mathematical operations.

**Modifiers:**onlyOwner: Restricts certain functions to be callable only by the contract owner.

**Constructor:**Parameters: supplierAddress (address), totalSupply (uint256)

* Initializes the token supply by minting the specified totalSupply number of tokens to the supplierAddress.

**Public and External Functions:**

**lock**Parameters: \_reason (bytes32), \_amount (uint256), \_time (uint256)

* Locks a specified amount of tokens against the caller's address for a specific reason and duration.
* Checks if tokens are already locked for the given reason and requires a non-zero amount.
* Transfers the specified amount of tokens from the caller to the contract and updates the locked token information.

**tokensLocked**Parameters: \_account (address), \_reason (bytes32)

* Returns the number of tokens locked for a specified address and reason.

**tokensLockedAtTime**Parameters: \_account (address), \_reason (bytes32), \_time (uint256)

* Returns the number of tokens locked for a specified address and reason at a specific time.

**totalBalanceOf**Parameters: \_account (address)

* Returns the total balance of tokens held by an address, including both transferable and locked tokens.

**extendLock**Parameters: \_reason (bytes32), \_time (uint256)

* Extends the lock duration for a specified reason and time.

**increaseLockAmount**Parameters: \_reason (bytes32), \_amount (uint256)

* Increases the number of tokens locked for a specified reason.

**tokensUnlockable**Parameters: \_account (address), \_reason (bytes32)

* Returns the number of tokens unlockable for a specified address and reason.

**Unlock**Parameters: \_reason (bytes32)

* Unlocks the unlockable tokens of the caller's address, transferring them back to the caller.

**getUnlockableTokens**Parameters: \_account (address)

* Calculates the total number of unlockable tokens for a specified address.

**Events:**

TokensLocked: Emitted when tokens are locked against an address for a specific reason and duration.

TokensUnlocked: Emitted when tokens are unlocked and transferred back to the caller's address.

**Ownership:**

The contract owner is defined during deployment and cannot be changed.

Certain functions are restricted to be callable only by the contract owner using the onlyOwner modifier.

This contract provides token holders with the ability to lock and manage their tokens for specified durations and reasons. They can check the locked and unlockable token balances, extend lock durations, increase locked token amounts, and unlock tokens when they become unlockable. The contract ensures the safety of token transfers and provides a convenient interface for managing token locking functionality.