SOLIDITY.*FINANCE*

AXIA Protocol - Smart Contract Audit Report

SUMMARY

AXIA Protocol is a Decentralized platform for Cryptocurrency Index Fund management which presents cryptocurrency enthusiasts/investors with opportunities of one-time investments in baskets of cryptocurrencies (Axia Funds) saving them time and energy that would have been spent sifting through thousands of tokens in search of which ones to buy for a portfolio. With Axia Funds, investors can be rest assured that their portfolio will only grow in profits over time.

AXIA Protocol consists of 7 Smart Contracts: The Protocol's token

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

> *Additional features included in the contract:*
>
> - *The token contract contains no mint function, so the supply of the token shall never increase. There is a burn function which could decrease the token's total supply.*
> - *Ownership/Adminship - Some functions are protected and can only be called by the contract Owner/Administrator.*
> - *Owners have the ability to update the rate variables for pools.*
> - *Utilization of SafeMath to prevent overflows.*
>
> *Audit Findings Summary*
>
> - *Overall, no serious security issues were identified.*
> - *The contract uses an* `assert` *check instead of a* `require` *check in the* `mulDiv()` *and* `scaledToken()` *functions. This is out of line with Solidity best practices, but in this case the usage poses no issue to the integrity of the contract.*
> - *The* `_minStakeAmount()`, `poolconfigs()`, `unstakeburnrate()` *and* `poolpercentages()` *functions are technically vulnerable to an arithmetic overflow, but it entirely in the control of the contract owner and thus no security issue is present.*

COMBINED AUDIT RESULTS

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

> *Date: October 22nd, 2020*

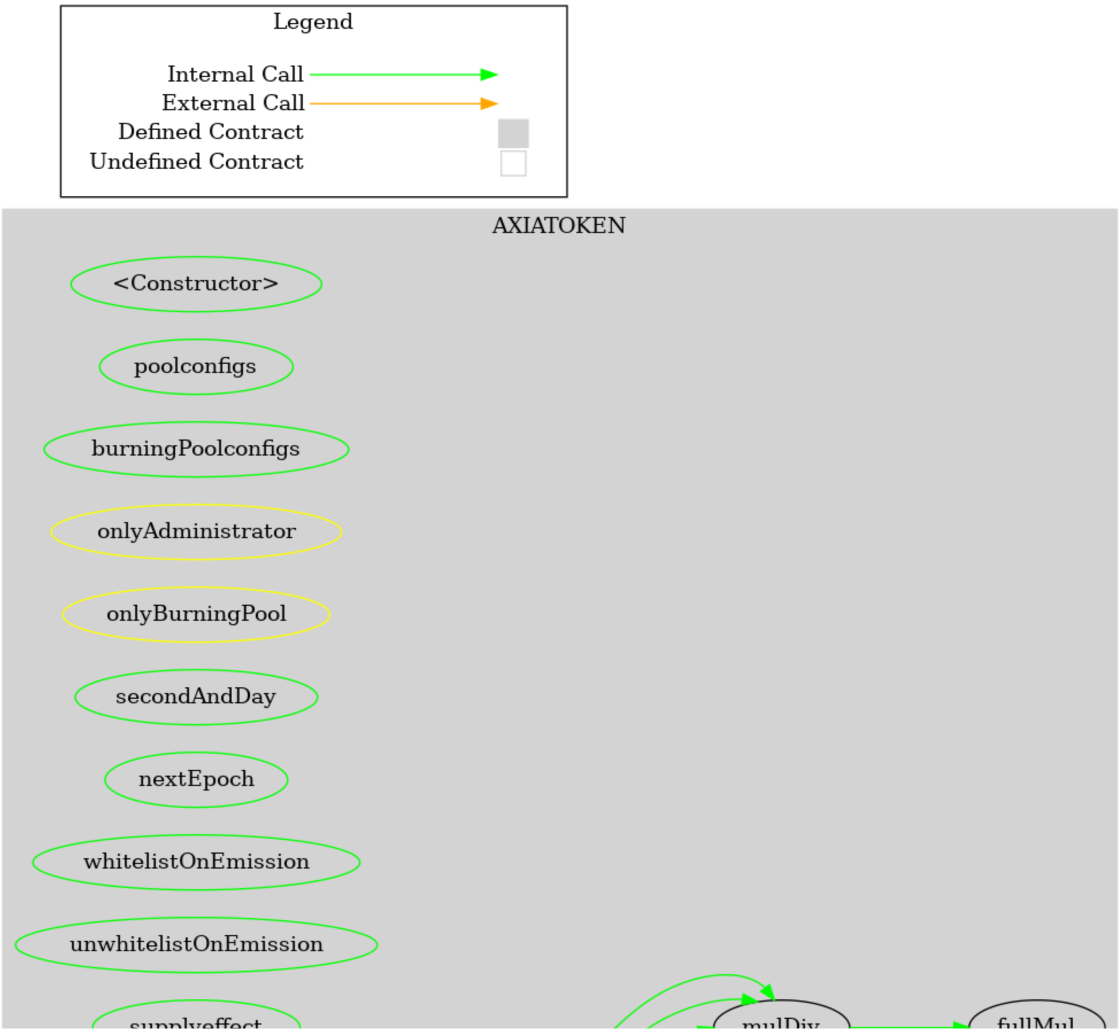
Vulnerability Category	Notes	Result
Arbitrary Storage Write	N/A	PASS
Arbitrary Jump	N/A	PASS
Delegate Call to Untrusted Contract	N/A	PASS
Dependence on Predictable Variables	N/A	PASS
Deprecated Opcodes	N/A	PASS
Ether Thief	N/A	PASS
Exceptions	N/A	PASS
External Calls	N/A	PASS

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

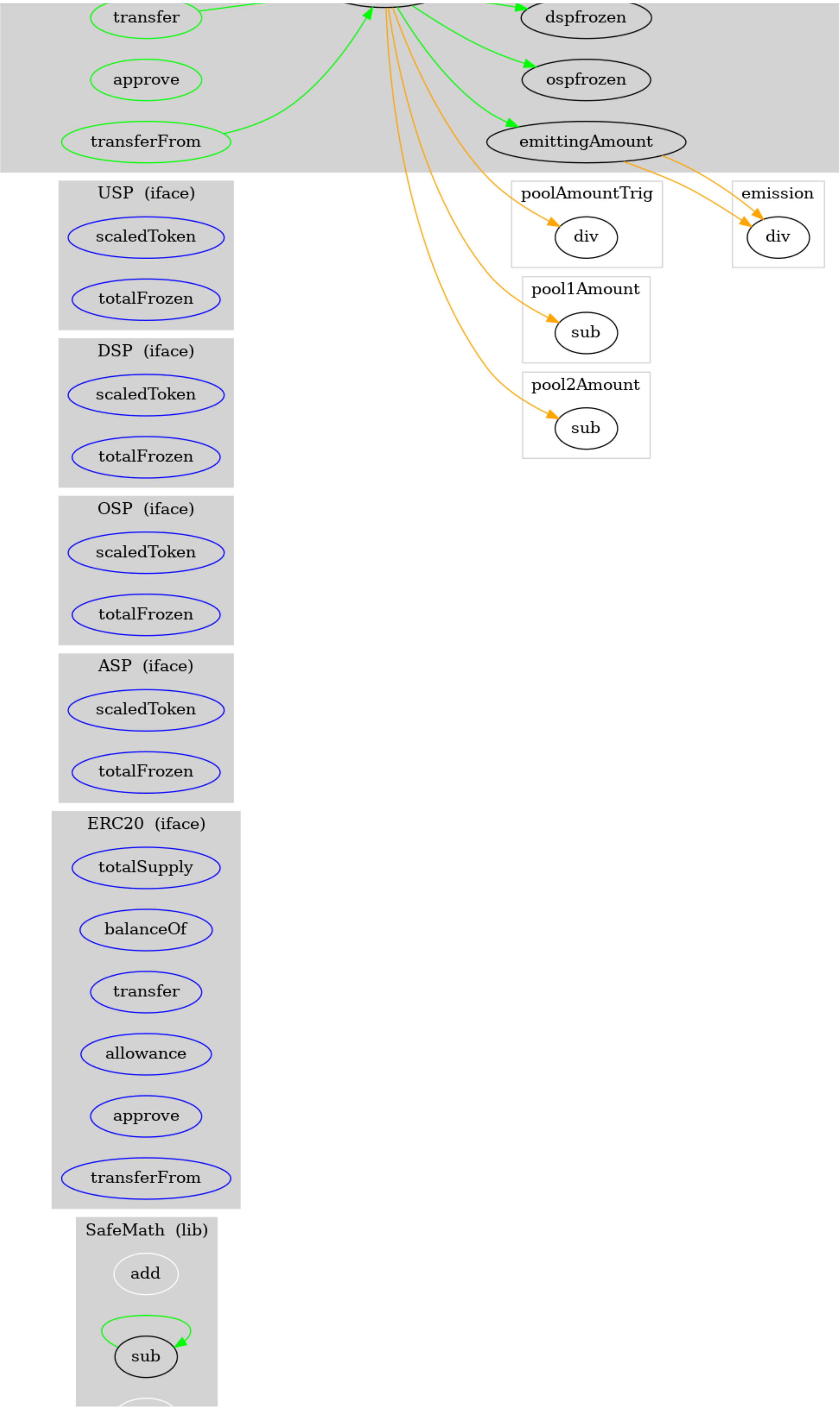
Vulnerability Category	Notes	Result
Integer Over/Underflow	N/A	PASS
Multiple Sends	N/A	PASS

DETAILS: AXIAV3 TOKEN

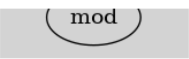
FUNCTION GRAPH



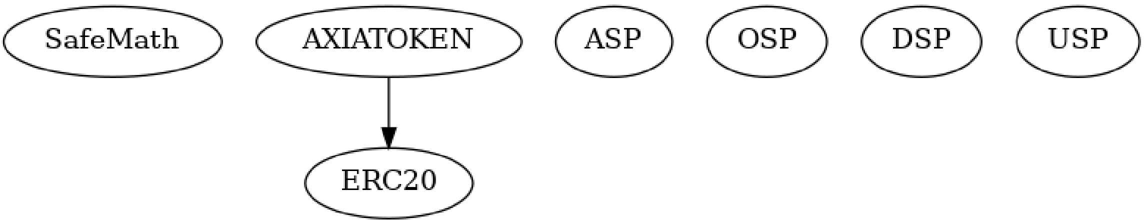
Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.



Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.



INHERITENCE CHART



FUNCTIONS OVERVIEW

```
($) = payable function
# = non-constant function

Int = Internal
Ext = External
Pub = Public

+ [Lib] SafeMath
   - [Int] add
   - [Int] sub
   - [Int] sub
   - [Int] mul
   - [Int] div
   - [Int] div
   - [Int] mod
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    - [Ext] totalSupply
    - [Ext] balanceOf
    - [Ext] transfer #
    - [Ext] allowance
    - [Ext] approve #
    - [Ext] transferFrom #

 + [Int] ASP
    - [Ext] scaledToken #
    - [Ext] totalFrozen

 + [Int] OSP
    - [Ext] scaledToken #
    - [Ext] totalFrozen

 + [Int] DSP
    - [Ext] scaledToken #
    - [Ext] totalFrozen

 + [Int] USP
    - [Ext] scaledToken #
    - [Ext] totalFrozen

 +  AXIATOKEN (ERC20)
    - [Pub]  #
    - [Pub] poolconfigs #
       - modifiers: onlyAdministrator
    - [Pub] burningPoolconfigs #
       - modifiers: onlyAdministrator
    - [Pub] secondAndDay #
       - modifiers: onlyAdministrator
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
      - modifiers: onlyAdministrator
   - [Pub] unwhitelistOnEmission #
      - modifiers: onlyAdministrator
   - [Pub] supplyeffect #
      - modifiers: onlyBurningPool
   - [Pub] poolpercentages #
      - modifiers: onlyAdministrator
   - [Pub] Burn #
   - [Pub] transfer #
   - [Pub] approve #
   - [Pub] transferFrom #
   - [Prv] _transfer #
   - [Int] emittingAmount #
   - [Pub] ospfrozen
   - [Pub] dspfrozen
   - [Pub] uspfrozen
   - [Pub] aspfrozen
   - [Pub] mulDiv
   - [Prv] fullMul
```

SOURCE CODE

Click here to download the source code as a .sol file.

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
library SafeMath {
    /**
     * @dev Returns the addition of two unsigned int
     * overflow.
     *
     * Counterpart to Solidity's `+` operator.
     *
     * Requirements:
     *
     * - Addition cannot overflow.
     */
    function add(uint256 a, uint256 b) internal pure
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow

        return c;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    /**
     * @dev Returns the subtraction of two unsigned
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
     *
     * Requirements:
     *
     * - Subtraction cannot overflow.
     */
    function sub(uint256 a, uint256 b) internal pure
        return sub(a, b, "SafeMath: subtraction over
    }

    /**
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * overflow (when the result is negative).

     *

     * Counterpart to Solidity's `-` operator.

     *

     * Requirements:

     *

     * - Subtraction cannot overflow.

     */

    function sub(uint256 a, uint256 b, string memory

        require(b <= a, errorMessage);

        uint256 c = a - b;



        return c;

    }



    /**
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * overflow.
     *
     * Counterpart to Solidity's `*` operator.
     *
     * Requirements:
     *
     * - Multiplication cannot overflow.
     */
    function mul(uint256 a, uint256 b) internal pure
        // Gas optimization: this is cheaper than re
        // benefit is lost if 'b' is also tested.
        // See: https://github.com/OpenZeppelin/open
        if (a == 0) {
            return 0;
        }

```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(c / a == b, "SafeMath: multiplicatio

        return c;
    }

    /**
     * @dev Returns the integer division of two unsi
     * division by zero. The result is rounded towar
     *
     * Counterpart to Solidity's `/` operator. Note:
     * `revert` opcode (which leaves remaining gas u
     * uses an invalid opcode to revert (consuming a
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function div(uint256 a, uint256 b) internal pure
        return div(a, b, "SafeMath: division by zero
    }

    /**
     * @dev Returns the integer division of two unsi
     * division by zero. The result is rounded towar
     *
     * Counterpart to Solidity's `/` operator. Note:
     * `revert` opcode (which leaves remaining gas u
     * uses an invalid opcode to revert (consuming a
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(b > 0, errorMessage);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is n

        return c;
    }

    /**
     * @dev Returns the remainder of dividing two un
     * Reverts when dividing by zero.
     *
     * Counterpart to Solidity's `%` operator. This
     * opcode (which leaves remaining gas untouched)
     * invalid opcode to revert (consuming all remai
     *
     * Requirements:
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * - The divisor cannot be zero.
     */
    function mod(uint256 a, uint256 b) internal pure
        return mod(a, b, "SafeMath: modulo by zero")
    }

    /**
     * @dev Returns the remainder of dividing two un
     * Reverts with custom message when dividing by
     *
     * Counterpart to Solidity's `%` operator. This
     * opcode (which leaves remaining gas untouched)
     * invalid opcode to revert (consuming all remai
     *
     * Requirements:
     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     */

    function mod(uint256 a, uint256 b, string memory

        require(b != 0, errorMessage);

        return a % b;

    }

}



//ERC20 Interface

interface ERC20 {

    function totalSupply() external view returns (ui

    function balanceOf(address account) external vie

    function transfer(address, uint) external return

    function allowance(address owner, address spende

    function approve(address, uint) external returns

    function transferFrom(address, address, uint) ex

    event Transfer(address indexed from, address ind
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    }



interface ASP {



   function scaledToken(uint amount) external return

   function totalFrozen() external view returns (uin

 }



interface OSP {



   function scaledToken(uint amount) external return

   function totalFrozen() external view returns (uin

 }



interface DSP {

```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
   function totalFrozen() external view returns (uin

 }



interface USP {



   function scaledToken(uint amount) external return

   function totalFrozen() external view returns (uin

 }



//========================================AXIA CONTRAC

contract AXIATOKEN is ERC20 {



    using SafeMath for uint256;



//========================================AXIA EVENTS=

```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        event NewDay(uint epoch, uint day, uint nextday)
        event BurnEvent(address indexed pool, address in
        event emissions(address indexed root, address in
        event TrigRewardEvent(address indexed root, addr
        event BasisPointAdded(uint value);



       // ERC-20 Parameters
        string public name;
        string public symbol;
        uint public decimals;
        uint public startdecimal;
        uint public override totalSupply;
        uint public initialsupply;

         //======================================STAKING
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    address public lonePool;

    address public swapPool;

    address public DefiPool;

    address public OraclePool;



    address public burningPool;



    uint public pool1Amount;

    uint public pool2Amount;

    uint public pool3Amount;

    uint public pool4Amount;

    uint public basisAmount;

    uint public poolAmountTrig;



    uint public TrigAmount;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    // ERC-20 Mappings
    mapping(address => uint) public override balance
    mapping(address => mapping(address => uint)) pub



    // Public Parameters
    uint crypto;
    uint startcrypto;
    uint public emission;
    uint public currentEpoch;
    uint public currentDay;
    uint public daysPerEpoch;
    uint public secondsPerDay;
    uint public genesis;
    uint public nextEpochTime;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    uint public amountToEmit;
    uint public BPE;

    //=======================================BASIS PO
    uint public bpValue;
    uint public actualValue;
    uint public TrigReward;
    uint public burnAmount;
    address administrator;
    uint totalEmitted;

    uint256 public pool1percentage = 500;
    uint256 public pool2percentage = 4500;
    uint256 public pool3percentage = 2500;
    uint256 public pool4percentage = 2500;
    uint256 public basispercentage = 500;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
address public messagesender;

// Public Mappings

mapping(address=>bool) public emission_Whitelist


//=====================================CREATION=
// Constructor
constructor() public {
    name = "AXIA TOKEN (axiaprotocol.io)";
    symbol = "AXIAv3";
    decimals = 18;
    startdecimal = 16;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        startcrypto =  1*10**startdecimal;
        totalSupply = 3800000*crypto;
        initialsupply = 120000000*startcrypto;
        emission = 7200*crypto;
        currentEpoch = 1;
        currentDay = 1;
        genesis = now;

        daysPerEpoch = 180;
        secondsPerDay = 86400;

        administrator = msg.sender;
        balanceOf[administrator] = initialsupply;
        emit Transfer(administrator, address(this),
        nextEpochTime = genesis + (secondsPerDay * d
        nextDayTime = genesis + secondsPerDay;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        emission_Whitelisted[administrator] = true;




    }


//=======================================CONFIGURAT


    function poolconfigs(address _axia, address _swa


        lonePool = _axia;

        swapPool = _swap;

        DefiPool = _defi;

        OraclePool = _oracle;


```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        return true;

    }



    function burningPoolconfigs(address _pooladdress



        burningPool = _pooladdress;



        return true;

    }




    modifier onlyAdministrator() {

        require(msg.sender == administrator, "Ownabl

        _;

    }

```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(msg.sender == burningPool, "Authoriz
        _;
    }


    function secondAndDay(uint _secondsperday, uint
       secondsPerDay = _secondsperday;
       daysPerEpoch = _daysperepoch;
        return true;
    }


    function nextEpoch(uint _nextepoch) public onlyA
       nextEpochTime = _nextepoch;


        return true;
    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
       emission_Whitelisted[_address] = true;
        return true;
    }

    function unwhitelistOnEmission(address _address)
       emission_Whitelisted[_address] = false;
        return true;
    }


    function supplyeffect(uint _amount) public onlyB
       totalSupply -= _amount;
       emit BurnEvent(burningPool, address(0x0), _am
        return true;
    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
   pool1percentage = _p1;

   pool2percentage = _p2;

   pool3percentage = _p3;

   pool4percentage = _p4;

   basispercentage = _basispercent;

   trigRewardpercentage = trigRe;



   return true;

}



function Burn(uint _amount) public returns (bool



   require(balanceOf[msg.sender] >= _amount, "Yo

   balanceOf[msg.sender] -= _amount;

   totalSupply -= _amount;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        return true;



     }



    //==========================================ERC20==

     // ERC20 Transfer function

     function transfer(address to, uint value) public

         _transfer(msg.sender, to, value);

         return true;

     }

     // ERC20 Approve function

     function approve(address spender, uint value) pu

         allowance[msg.sender][spender] = value;

         emit Approval(msg.sender, spender, value);

         return true;

     }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function transferFrom(address from, address to,

        require(value <= allowance[from][msg.sender]

        allowance[from][msg.sender] -= value;

        _transfer(from, to, value);

        return true;

    }





    // Internal transfer function which includes the

    function _transfer(address _from, address _to, u


        messagesender = msg.sender; //this is the pe




        require(balanceOf[_from] >= _value, 'Must no
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        balanceOf[_from] -= _value;



        if(emission_Whitelisted[messagesender] == fa

                if(now >= nextDayTime){

                amountToEmit = emittingAmount();



                uint basisAmountQuota = mulDiv(amoun
                amountToEmit = amountToEmit - basisA
                basisAmount = basisAmountQuota;

                pool1Amount = mulDiv(amountToEmit, p
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
                pool3Amount = mulDiv(amountToEmit, p

                pool4Amount = mulDiv(amountToEmit, p






                poolAmountTrig = mulDiv(amountToEmit

                TrigAmount = poolAmountTrig.div(2);


                pool1Amount = pool1Amount.sub(TrigAm

                pool2Amount = pool2Amount.sub(TrigAm


                TrigReward = poolAmountTrig;



                uint Ofrozenamount = ospfrozen();

                uint Dfrozenamount = dspfrozen();

                uint Ufrozenamount = uspfrozen();
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
                balanceOf[address(this)] += basisAmo
                emit Transfer(address(this), address
                BPE += basisAmount;



                if(Ofrozenamount > 0){


                OSP(OraclePool).scaledToken(pool4Amo
                balanceOf[OraclePool] += pool4Amount
                emit Transfer(address(this), OracleP




                }else{
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
 emit Transfer(address(this), addres

 BPE += pool4Amount;

}

if(Dfrozenamount > 0){

DSP(DefiPool).scaledToken(pool3Amoun
balanceOf[DefiPool] += pool3Amount;
emit Transfer(address(this), DefiPoo

}else{
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
 emit Transfer(address(this), addres

 BPE += pool3Amount;



}



if(Ufrozenamount > 0){


USP(swapPool).scaledToken(pool2Amoun

balanceOf[swapPool] += pool2Amount;

emit Transfer(address(this), swapPoo




}else{



 balanceOf[address(this)] += pool2Am

 emit Transfer(address(this), addres
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
}

if(Afrozenamount > 0){

 ASP(lonePool).scaledToken(pool1Amou
 balanceOf[lonePool] += pool1Amount;
 emit Transfer(address(this), lonePo

}else{

 balanceOf[address(this)] += pool1Am
 emit Transfer(address(this), addres
 BPE += pool1Amount;

}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
                 nextDayTime += secondsPerDay;

                 currentDay += 1;

                 emit NewDay(currentEpoch, currentDay


                 //reward the wallet that triggered t

                 balanceOf[_from] += TrigReward; //th

                 emit Transfer(address(this), _from,

                 emit TrigRewardEvent(address(this),


             }




         }


        balanceOf[_to] += _value;

        emit Transfer(_from, _to, _value);
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
//======================================EMISSION

// Internal - Update emission function

function emittingAmount() internal returns(uint)

    if(now >= nextEpochTime){

        currentEpoch += 1;

        if(currentEpoch > 10){
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
           BPE -= emission.div(2);

           balanceOf[address(this)] -= emission.



        }

        emission = emission/2;

        nextEpochTime += (secondsPerDay * daysPe

        emit NewEpoch(currentEpoch, emission, ne


    }


    return emission;



}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
function ospfrozen() public view returns(uint){
    return OSP(OraclePool).totalFrozen();
}
function dspfrozen() public view returns(uint){
    return DSP(DefiPool).totalFrozen();
}
function uspfrozen() public view returns(uint){
    return USP(swapPool).totalFrozen();
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        }



        function aspfrozen() public view returns(uint){



            return ASP(lonePool).totalFrozen();



        }



         function mulDiv (uint x, uint y, uint z) public

              (uint l, uint h) = fullMul (x, y);

              assert (h < z);

              uint mm = mulmod (x, y, z);

              if (mm > l) h -= 1;

              l -= mm;

              uint pow2 = z & -z;

              z /= pow2;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

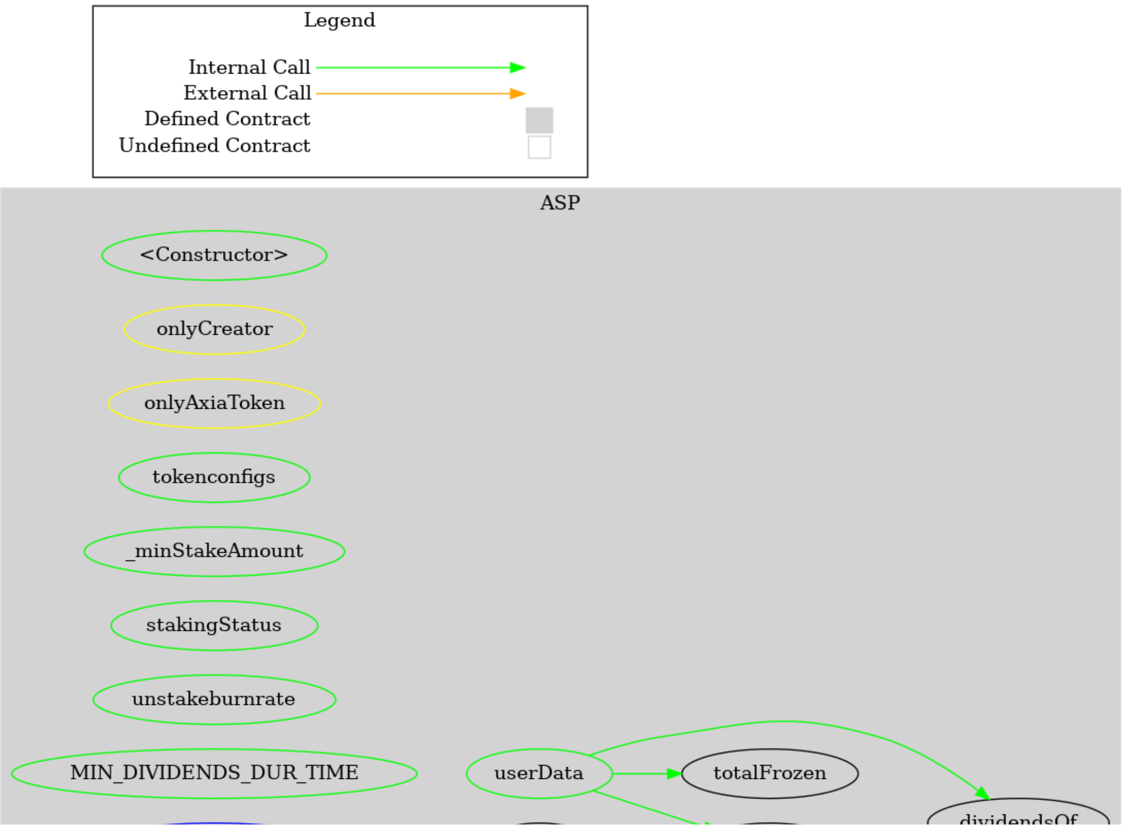
```
        l += h * ((-pow2) / pow2 + 1);

        uint r = 1;

        r *= 2 - z * r;

        r *= 2 - z * r;

        r *= 2 - z * r;

        r *= 2 - z * r;

        r *= 2 - z * r;

        r *= 2 - z * r;

        r *= 2 - z * r;

        r *= 2 - z * r;

        return l * r;

  }



   function fullMul (uint x, uint y) private pure

        uint mm = mulmod (x, y, uint (-1));

        l = x * y;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

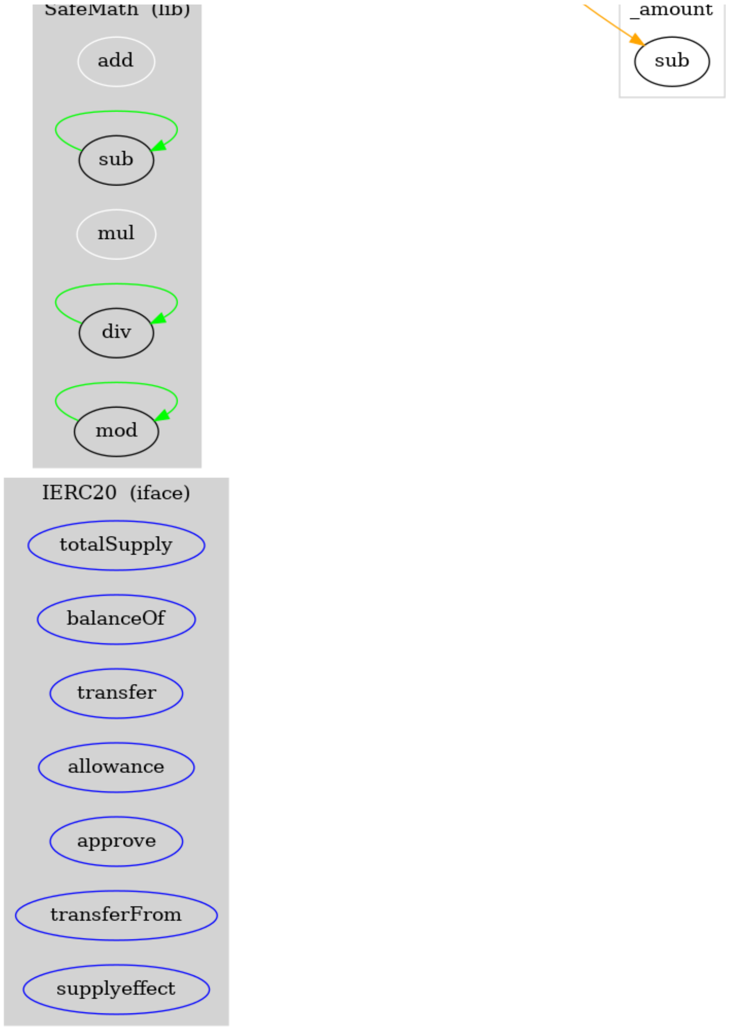
```
                if (mm < 1) h -= 1;


        }









    }
```

DETAILS: AXIA STAKING POOL

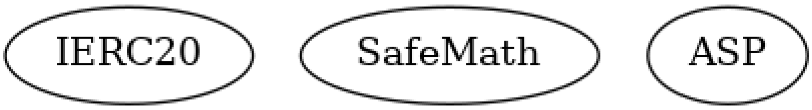
FUNCTION GRAPH



Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.



INHERITENCE CHART



FUNCTIONS OVERVIEW

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
Int = Internal
Ext = External
Pub = Public

+ [Int] IERC20
   - [Ext] totalSupply
   - [Ext] balanceOf
   - [Ext] transfer #
   - [Ext] allowance
   - [Ext] approve #
   - [Ext] transferFrom #
   - [Ext] supplyeffect #

+ [Lib] SafeMath
   - [Int] add
   - [Int] sub
   - [Int] sub
   - [Int] mul
   - [Int] div
   - [Int] div
   - [Int] mod
   - [Int] mod

+  ASP
   - [Pub]  #
   - [Pub] tokenconfigs #
      - modifiers: onlyCreator
   - [Pub] _minStakeAmount #
      - modifiers: onlyCreator
   - [Pub] stakingStatus #
      - modifiers: onlyCreator
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
         - modifiers: onlyCreator
      - [Ext] StakeAxiaTokens #
      - [Ext] UnstakeAxiaTokens #
      - [Pub] totalFrozen
      - [Pub] frozenOf
      - [Pub] dividendsOf
      - [Pub] userData
      - [Int] _stake #
      - [Int] _unstake #
      - [Pub] TakeDividends #
      - [Ext] scaledToken #
         - modifiers: onlyAxiaToken
      - [Pub] mulDiv
      - [Prv] fullMul
```

SOURCE CODE

Click here to download the source code as a .sol file.

```
/**

 *Submitted for verification at Etherscan.io on 2020

*/
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
/*


* @dev This is the Axia Protocol Staking pool contra

* when they make stakes of Axia tokens.


* stakers reward come from the daily emission from t

* this happens daily and upon the reach of a new epo

* halvings are experienced on the emitting amount of



* on the 11th epoch all the tokens would have been c

* from here on, the stakers will still be earning fr

* which would now be coming from the accumulated bas


* upon unstaking, stakers are charged a fee of 1% of

* burnt forever, thereby reducing the total supply.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
*/

pragma solidity 0.6.4;



interface IERC20 {


    function totalSupply() external view returns (ui


    function balanceOf(address account) external vie


    function transfer(address recipient, uint256 amo


    function allowance(address owner, address spende


    function approve(address spender, uint256 amount
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function transferFrom(address sender, address re

    function supplyeffect(uint _amount) external ret

    event Transfer(address indexed from, address ind

    event Approval(address indexed owner, address in

}



library SafeMath {

    /**

     * @dev Returns the addition of two unsigned int

     * overflow.

     *

     * Counterpart to Solidity's `+` operator.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * Requirements:
     *
     * - Addition cannot overflow.
     */
    function add(uint256 a, uint256 b) internal pure
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow

        return c;
    }

    /**
     * @dev Returns the subtraction of two unsigned
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * Requirements:
     *
     * - Subtraction cannot overflow.
     */
    function sub(uint256 a, uint256 b) internal pure
        return sub(a, b, "SafeMath: subtraction over
    }

    /**
     * @dev Returns the subtraction of two unsigned
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
     *
     * Requirements:
     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     */

    function sub(uint256 a, uint256 b, string memory

        require(b <= a, errorMessage);

        uint256 c = a - b;



        return c;

    }



    /**

     * @dev Returns the multiplication of two unsign

     * overflow.

     *

     * Counterpart to Solidity's `*` operator.

     *

     * Requirements:

     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     */
    function mul(uint256 a, uint256 b) internal pure
        // Gas optimization: this is cheaper than re
        // benefit is lost if 'b' is also tested.
        // See: https://github.com/OpenZeppelin/open
        if (a == 0) {
            return 0;
        }

        uint256 c = a * b;
        require(c / a == b, "SafeMath: multiplicatio

        return c;
    }

    /**
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * division by zero. The result is rounded towar

     *

     * Counterpart to Solidity's `/` operator. Note:

     * `revert` opcode (which leaves remaining gas u

     * uses an invalid opcode to revert (consuming a

     *

     * Requirements:

     *

     * - The divisor cannot be zero.

     */

    function div(uint256 a, uint256 b) internal pure

        return div(a, b, "SafeMath: division by zero

    }



    /**

     * @dev Returns the integer division of two unsi
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     *
     * Counterpart to Solidity's `/` operator. Note:
     * `revert` opcode (which leaves remaining gas u
     * uses an invalid opcode to revert (consuming a
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function div(uint256 a, uint256 b, string memory
        require(b > 0, errorMessage);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is n

        return c;
    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
/**
 * @dev Returns the remainder of dividing two un
 * Reverts when dividing by zero.
 *
 * Counterpart to Solidity's `%` operator. This
 * opcode (which leaves remaining gas untouched)
 * invalid opcode to revert (consuming all remai
 *
 * Requirements:
 *
 * - The divisor cannot be zero.
 */
function mod(uint256 a, uint256 b) internal pure
    return mod(a, b, "SafeMath: modulo by zero")
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * @dev Returns the remainder of dividing two un
     * Reverts with custom message when dividing by
     *
     * Counterpart to Solidity's `%` operator. This
     * opcode (which leaves remaining gas untouched)
     * invalid opcode to revert (consuming all remai
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function mod(uint256 a, uint256 b, string memory
        require(b != 0, errorMessage);
        return a % b;
    }
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
contract ASP{

    using SafeMath for uint256;

    //======================================EVENTS==
    event StakeEvent(address indexed staker, address
    event UnstakeEvent(address indexed unstaker, add
    event RewardEvent(address indexed staker, addres
    event RewardStake(address indexed staker, addres


    //======================================STAKING
    address public Axiatoken;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
uint256 constant private FLOAT_SCALAR = 2**64;

uint256 public MINIMUM_STAKE = 10000000000000000

    uint256 public MIN_DIVIDENDS_DUR = 18 hours;

    uint256 private  UNSTAKE_FEE = 1; //1% burns

    uint public infocheck;

    uint _burnedAmount;

    uint actualValue;



struct User {

            uint256 balance;

            uint256 frozen;

            int256 scaledPayout;

            uint256 staketime;

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
            uint256 totalSupply;

            uint256 totalFrozen;

            mapping(address => User) users;

            uint256 scaledPayoutPerToken; //pool

            address admin;

    }



    Info private info;





    constructor() public {



        info.admin = msg.sender;

            stakingEnabled = false;



    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
//=========================================ADMINSTRATIO

        modifier onlyCreator() {
        require(msg.sender == info.admin, "Ownable:
        _;
    }

    modifier onlyAxiaToken() {
        require(msg.sender == Axiatoken, "Authorizat
        _;
    }

         function tokenconfigs(address _axiatoken) p
        Axiatoken = _axiatoken;
        return true;
    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        function _minStakeAmount(uint256 _number) on

                MINIMUM_STAKE = _number*100000000000

        }

    function stakingStatus(bool _status) public only
    require(Axiatoken != address(0), "Pool address i
        stakingEnabled = _status;
    }

    function unstakeburnrate(uint _rate) public only
        UNSTAKE_FEE = _rate;
        return true;
    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function MIN_DIVIDENDS_DUR_TIME(uint256 _minDura



        MIN_DIVIDENDS_DUR = _minDuration;



    }

//=====================================USER WRITE==



        function StakeAxiaTokens(uint256 _tokens) ex

                _stake(_tokens);

        }



    function UnstakeAxiaTokens(uint256 _tokens) exte

                _unstake(_tokens);

        }



//=====================================USER READ===
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function totalFrozen() public view returns (

            return info.totalFrozen;

    }



function frozenOf(address _user) public view ret

            return info.users[_user].frozen;

    }



    function dividendsOf(address _user) public v



        if(info.users[_user].staketime < MIN_DIV

            return 0;

        }else{

         return uint256(int256(info.scaledPayout

        }

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        function userData(address _user) public view
        returns (uint256 totalTokensFrozen, uint256
        uint256 userDividends, uint256 userStaketime

                return (totalFrozen(), frozenOf(_use



        }




//======================================ACTION CALLS


        function _stake(uint256 _amount) internal {


            require(stakingEnabled, "Staking not yet
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(IERC20(Axiatoken).balanceOf(
        require(frozenOf(msg.sender) + _amou
        require(IERC20(Axiatoken).allowance(


        info.users[msg.sender].staketime = n
        info.totalFrozen += _amount;
        info.users[msg.sender].frozen += _am


        info.users[msg.sender].scaledPayout 
        IERC20(Axiatoken).transferFrom(msg.s


emit StakeEvent(msg.sender, address(this), _
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
                require(frozenOf(msg.sender) >= _amo


                info.totalFrozen -= _amount;

                info.users[msg.sender].frozen -= _am

                info.users[msg.sender].scaledPayout


                _burnedAmount = mulDiv(_amount, UNST

                actualValue = _amount.sub(_burnedAmo


                require(IERC20(Axiatoken).transfer(m

        emit UnstakeEvent(address(this), msg.sender,




                require(IERC20(Axiatoken).transfer(a

                IERC20(Axiatoken).supplyeffect(_burn
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        TakeDividends();

}

function TakeDividends() public returns (uin

        uint256 _dividends = dividendsOf(msg
        require(_dividends >= 0, "you do not
        info.users[msg.sender].scaledPayout

        require(IERC20(Axiatoken).transfer(m
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
                return _dividends;

        }

        function scaledToken(uint _amount) external

                info.scaledPayoutPerToken += _amount
                infocheck = info.scaledPayoutPerToke
                return true;

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
(uint l, uint h) = fullMul (x, y);
assert (h < z);
uint mm = mulmod (x, y, z);
if (mm > l) h -= 1;
l -= mm;
uint pow2 = z & -z;
z /= pow2;
l /= pow2;
l += h * ((-pow2) / pow2 + 1);
uint r = 1;
r *= 2 - z * r;
r *= 2 - z * r;
r *= 2 - z * r;
r *= 2 - z * r;
r *= 2 - z * r;
r *= 2 - z * r;
```

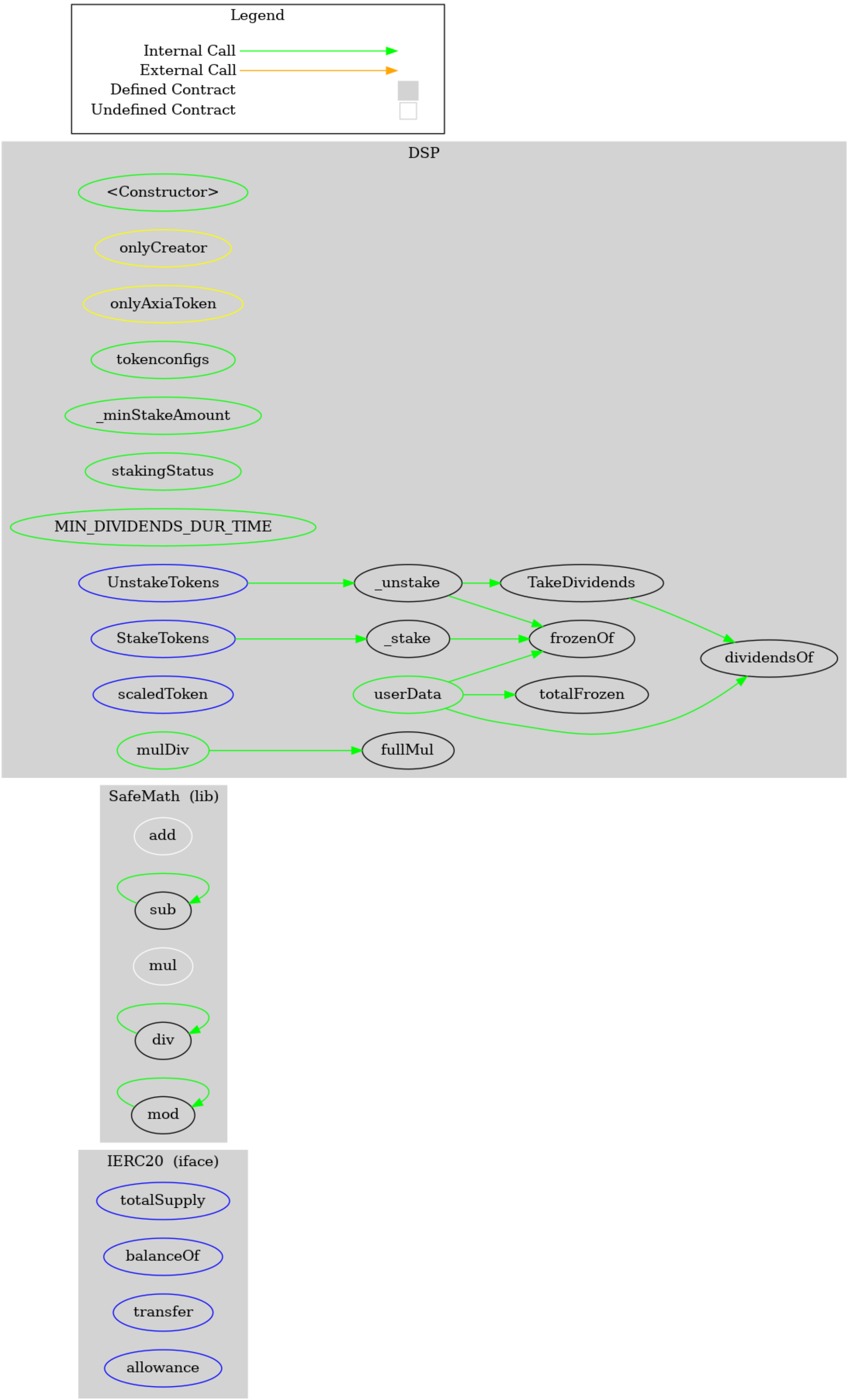
Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
              r *= 2 - z * r;

              return l * r;

    }



    function fullMul (uint x, uint y) private pure r

              uint mm = mulmod (x, y, uint (-1));

              l = x * y;

              h = mm - l;

              if (mm < l) h -= 1;

    }



}
```

DETAILS: DEFI STAKING POOL

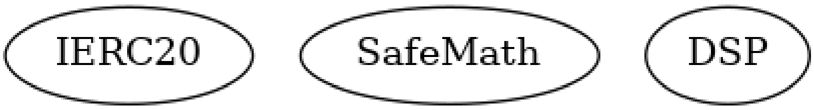
FUNCTION GRAPH

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.



Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

INHERITENCE CHART



FUNCTIONS OVERVIEW

```
($) = payable function
# = non-constant function

Int = Internal
Ext = External
Pub = Public

+ [Int] IERC20
   - [Ext] totalSupply
   - [Ext] balanceOf
   - [Ext] transfer #
   - [Ext] allowance
   - [Ext] approve #
   - [Ext] transferFrom #

+ [Lib] SafeMath
   - [Int] add
   - [Int] sub
   - [Int] sub
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
   - [Int] mod

+  DSP
   - [Pub]  #
   - [Pub] tokenconfigs #
      - modifiers: onlyCreator
   - [Pub] _minStakeAmount #
      - modifiers: onlyCreator
   - [Pub] stakingStatus #
      - modifiers: onlyCreator
   - [Pub] MIN_DIVIDENDS_DUR_TIME #
      - modifiers: onlyCreator
   - [Ext] StakeTokens #
   - [Ext] UnstakeTokens #
   - [Pub] totalFrozen
   - [Pub] frozenOf
   - [Pub] dividendsOf
   - [Pub] userData
   - [Int] _stake #
   - [Int] _unstake #
   - [Pub] TakeDividends #
   - [Ext] scaledToken #
      - modifiers: onlyAxiaToken
   - [Pub] mulDiv
   - [Prv] fullMul
```

SOURCE CODE

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
/**

 *Submitted for verification at Etherscan.io on 2020

*/



/*



* @dev This is the Axia Protocol Staking pool 2 cont

* a part of the protocol where stakers are rewarded

* when they make stakes of liquidity tokens from the



* stakers reward come from the daily emission from t

* this happens daily and upon the reach of a new epo

* halvings are experienced on the emitting amount of



* on the 11th epoch all the tokens would have been c
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
* stakers are not charged any fee for unstaking.



*/

pragma solidity 0.6.4;


interface IERC20 {


    function totalSupply() external view returns (ui


    function balanceOf(address account) external vie


    function transfer(address recipient, uint256 amo
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function approve(address spender, uint256 amount

    function transferFrom(address sender, address re

    event Transfer(address indexed from, address ind

    event Approval(address indexed owner, address in
}

library SafeMath {
    /**
     * @dev Returns the addition of two unsigned int
     * overflow.
     *
     * Counterpart to Solidity's `+` operator.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     *
     * - Addition cannot overflow.
     */
    function add(uint256 a, uint256 b) internal pure
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow

        return c;
    }

    /**
     * @dev Returns the subtraction of two unsigned
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     *
     * - Subtraction cannot overflow.
     */
    function sub(uint256 a, uint256 b) internal pure
        return sub(a, b, "SafeMath: subtraction over
    }

    /**
     * @dev Returns the subtraction of two unsigned
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
     *
     * Requirements:
     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function sub(uint256 a, uint256 b, string memory
        require(b <= a, errorMessage);
        uint256 c = a - b;

        return c;
    }


    /**
     * @dev Returns the multiplication of two unsign
     * overflow.
     *
     * Counterpart to Solidity's `*` operator.
     *
     * Requirements:
     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function mul(uint256 a, uint256 b) internal pure
        // Gas optimization: this is cheaper than re
        // benefit is lost if 'b' is also tested.
        // See: https://github.com/OpenZeppelin/open
        if (a == 0) {
            return 0;
        }

        uint256 c = a * b;
        require(c / a == b, "SafeMath: multiplicatio

        return c;
    }

    /**
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     *
     * Counterpart to Solidity's `/` operator. Note:
     * `revert` opcode (which leaves remaining gas u
     * uses an invalid opcode to revert (consuming a
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function div(uint256 a, uint256 b) internal pure
        return div(a, b, "SafeMath: division by zero
    }

    /**
     * @dev Returns the integer division of two unsi
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * Counterpart to Solidity's `/` operator. Note:
     * `revert` opcode (which leaves remaining gas u
     * uses an invalid opcode to revert (consuming a
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function div(uint256 a, uint256 b, string memory
        require(b > 0, errorMessage);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is n

        return c;
    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * @dev Returns the remainder of dividing two un

     * Reverts when dividing by zero.

     *

     * Counterpart to Solidity's `%` operator. This

     * opcode (which leaves remaining gas untouched)

     * invalid opcode to revert (consuming all remai

     *

     * Requirements:

     *

     * - The divisor cannot be zero.

     */

    function mod(uint256 a, uint256 b) internal pure

        return mod(a, b, "SafeMath: modulo by zero")

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * Reverts with custom message when dividing by
     *
     * Counterpart to Solidity's `%` operator. This
     * opcode (which leaves remaining gas untouched)
     * invalid opcode to revert (consuming all remai
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function mod(uint256 a, uint256 b, string memory
        require(b != 0, errorMessage);
        return a % b;
    }
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
contract DSP{


    using SafeMath for uint256;


//========================================EVENTS======

    event StakeEvent(address indexed staker, address

    event UnstakeEvent(address indexed unstaker, add

    event RewardEvent(address indexed staker, addres

    event RewardStake(address indexed staker, addres




//========================================STAKING POOL

    address public Axiatoken;

    address public DefiIndexFunds;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    bool public stakingEnabled;


    uint256 constant private FLOAT_SCALAR = 2**64;
    uint256 public MINIMUM_STAKE = 10000000000000000
        uint256 public MIN_DIVIDENDS_DUR = 18 hours;


        uint public infocheck;


    struct User {
                uint256 balance;
                uint256 frozen;
                int256 scaledPayout;
                uint256 staketime;
        }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        uint256 totalSupply;
        uint256 totalFrozen;
        mapping(address => User) users;
        uint256 scaledPayoutPerToken; //pool
        address admin;
}


Info private info;




constructor() public {


info.admin = msg.sender;
stakingEnabled = false;
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    modifier onlyCreator() {

    require(msg.sender == info.admin, "Ownable:

    _;

}



modifier onlyAxiaToken() {

    require(msg.sender == Axiatoken, "Authorizat

    _;

}



     function tokenconfigs(address _axiatoken, a

    require(_axiatoken != _defiindex, "Insertion

    require(_axiatoken != address(0) && _defiind

    Axiatoken = _axiatoken;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    }


        function _minStakeAmount(uint256 _number) on


                MINIMUM_STAKE = _number*100000000000


        }


        function stakingStatus(bool _status) public 
            require(Axiatoken != address(0) && DefiI
        stakingEnabled = _status;
    }



    function MIN_DIVIDENDS_DUR_TIME(uint256 _minDura
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    }


//======================================USER WRITE==


        function StakeTokens(uint256 _tokens) extern

                _stake(_tokens);

        }


        function UnstakeTokens(uint256 _tokens) exte

                _unstake(_tokens);

        }



//======================================USER READ===
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
            return info.totalFrozen;

    }



function frozenOf(address _user) public view ret

            return info.users[_user].frozen;

    }



    function dividendsOf(address _user) public v



        if(info.users[_user].staketime < MIN_DIV

            return 0;

        }else{

         return uint256(int256(info.scaledPayout

        }


```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        function userData(address _user) public view
        returns (uint256 totalTokensFrozen, uint256 
        uint256 userDividends, uint256 userStaketime

                return (totalFrozen(), frozenOf(_use


        }


//=====================================ACTION CALLS

        function _stake(uint256 _amount) internal {
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(IERC20(DefiIndexFunds).balan
        require(frozenOf(msg.sender) + _amou
        require(IERC20(DefiIndexFunds).allow

        info.users[msg.sender].staketime = n
        info.totalFrozen += _amount;
        info.users[msg.sender].frozen += _am

        info.users[msg.sender].scaledPayout 
        IERC20(DefiIndexFunds).transferFrom(

emit StakeEvent(msg.sender, address(this), _
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
function _unstake(uint256 _amount) internal

        require(frozenOf(msg.sender) >= _amo

        info.totalFrozen -= _amount;
        info.users[msg.sender].frozen -= _am
        info.users[msg.sender].scaledPayout

        require(IERC20(DefiIndexFunds).trans
emit UnstakeEvent(address(this), msg.sender,

        TakeDividends();
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
                uint256 _dividends = dividendsOf(msg

                require(_dividends >= 0, "you do not

                info.users[msg.sender].scaledPayout


                require(IERC20(Axiatoken).transfer(m

                emit RewardEvent(msg.sender, address


                return _dividends;




        }



    function scaledToken(uint _amount) external only


                info.scaledPayoutPerToken += _amount
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
}




function mulDiv (uint x, uint y, uint z) public

   (uint l, uint h) = fullMul (x, y);

   assert (h < z);

   uint mm = mulmod (x, y, z);

   if (mm > l) h -= 1;

   l -= mm;

   uint pow2 = z & -z;

   z /= pow2;

   l /= pow2;

   l += h * ((-pow2) / pow2 + 1);

   uint r = 1;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
          r *= 2 - z * r;

          r *= 2 - z * r;

          r *= 2 - z * r;

          r *= 2 - z * r;

          r *= 2 - z * r;

          r *= 2 - z * r;

          return l * r;

    }



     function fullMul (uint x, uint y) private pure

          uint mm = mulmod (x, y, uint (-1));

          l = x * y;

          h = mm - l;

          if (mm < l) h -= 1;

    }
```

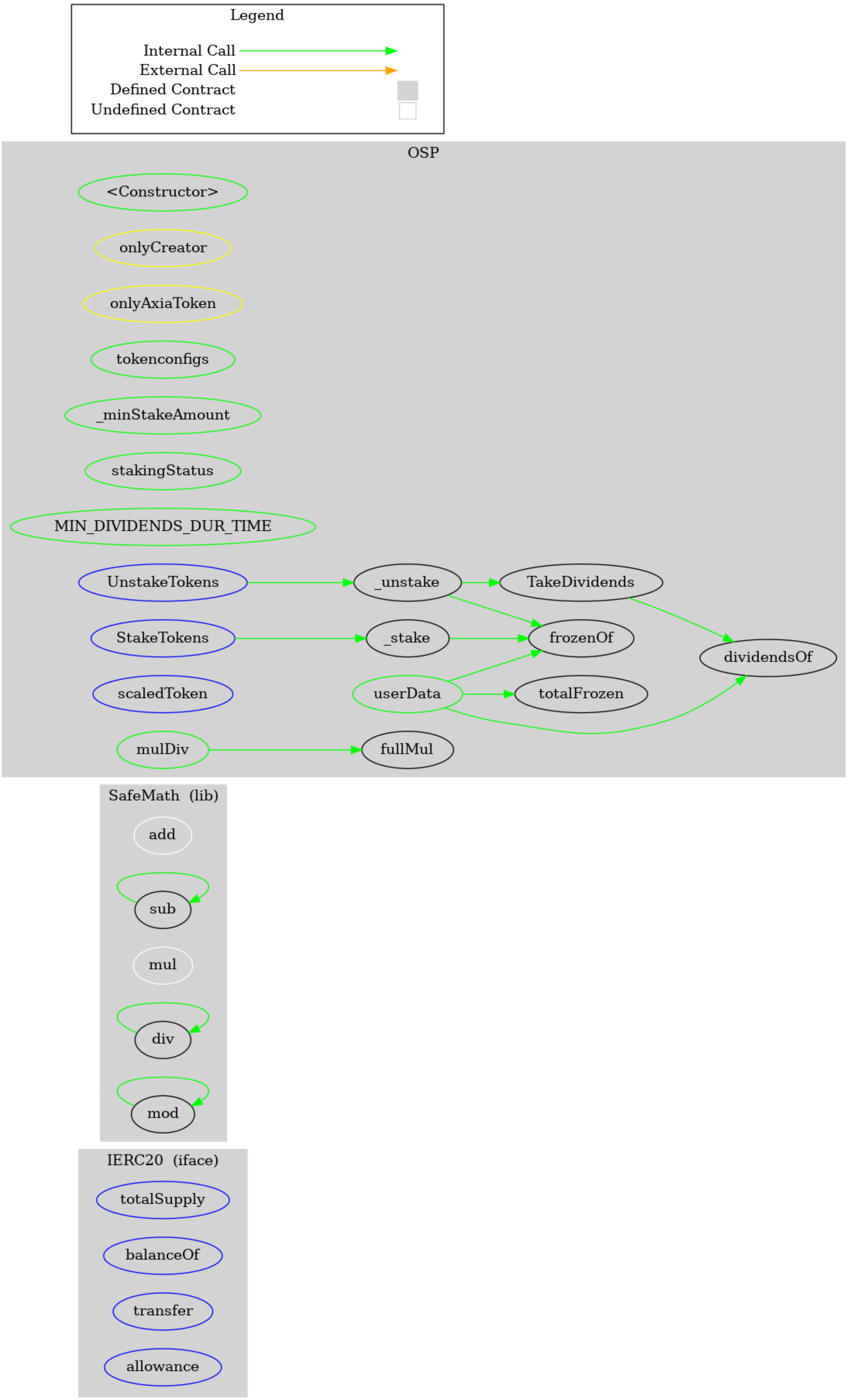
Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    }
```

DETAILS: ORACLE STAKING POOL

FUNCTION GRAPH

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.



Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

INHERITENCE CHART

IERC20
SafeMath
OSP

FUNCTIONS OVERVIEW

```
($) = payable function
# = non-constant function

Int = Internal
Ext = External
Pub = Public

+ [Int] IERC20
   - [Ext] totalSupply
   - [Ext] balanceOf
   - [Ext] transfer #
   - [Ext] allowance
   - [Ext] approve #
   - [Ext] transferFrom #

+ [Lib] SafeMath
   - [Int] add
   - [Int] sub
   - [Int] sub
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
   - [Int] mod

+  OSP
   - [Pub]  #
   - [Pub] tokenconfigs #
      - modifiers: onlyCreator
   - [Pub] _minStakeAmount #
      - modifiers: onlyCreator
   - [Pub] stakingStatus #
      - modifiers: onlyCreator
   - [Pub] MIN_DIVIDENDS_DUR_TIME #
      - modifiers: onlyCreator
   - [Ext] StakeTokens #
   - [Ext] UnstakeTokens #
   - [Pub] totalFrozen
   - [Pub] frozenOf
   - [Pub] dividendsOf
   - [Pub] userData
   - [Int] _stake #
   - [Int] _unstake #
   - [Pub] TakeDividends #
   - [Ext] scaledToken #
      - modifiers: onlyAxiaToken
   - [Pub] mulDiv
   - [Prv] fullMul
```

SOURCE CODE

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
/**

 *Submitted for verification at Etherscan.io on 2020

*/



/*



* @dev This is the Axia Protocol Staking pool 1 cont

* a part of the protocol where stakers are rewarded

* when they make stakes of liquidity tokens from the



* stakers reward come from the daily emission from t

* this happens daily and upon the reach of a new epo

* halvings are experienced on the emitting amount of



* on the 11th epoch all the tokens would have been c
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
* stakers are not charged any fee for unstaking.

*/

pragma solidity 0.6.4;

interface IERC20 {

    function totalSupply() external view returns (ui

    function balanceOf(address account) external vie

    function transfer(address recipient, uint256 amo

    function allowance(address owner, address spende
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function transferFrom(address sender, address re

    event Transfer(address indexed from, address ind

    event Approval(address indexed owner, address in
}

library SafeMath {
    /**
     * @dev Returns the addition of two unsigned int
     * overflow.
     *
     * Counterpart to Solidity's `+` operator.
     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * - Addition cannot overflow.
     */
    function add(uint256 a, uint256 b) internal pure
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow

        return c;
    }

    /**
     * @dev Returns the subtraction of two unsigned
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * - Subtraction cannot overflow.

     */

    function sub(uint256 a, uint256 b) internal pure

        return sub(a, b, "SafeMath: subtraction over

    }



    /**

     * @dev Returns the subtraction of two unsigned

     * overflow (when the result is negative).

     *

     * Counterpart to Solidity's `-` operator.

     *

     * Requirements:

     *

     * - Subtraction cannot overflow.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(b <= a, errorMessage);

        uint256 c = a - b;



        return c;

    }




    /**

     * @dev Returns the multiplication of two unsign

     * overflow.

     *

     * Counterpart to Solidity's `*` operator.

     *

     * Requirements:

     *

     * - Multiplication cannot overflow.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        // Gas optimization: this is cheaper than re
        // benefit is lost if 'b' is also tested.
        // See: https://github.com/OpenZeppelin/open
        if (a == 0) {
            return 0;
        }

        uint256 c = a * b;
        require(c / a == b, "SafeMath: multiplicatio

        return c;
    }

    /**
     * @dev Returns the integer division of two unsi
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * Counterpart to Solidity's `/` operator. Note:

     * `revert` opcode (which leaves remaining gas u

     * uses an invalid opcode to revert (consuming a

     *

     * Requirements:

     *

     * - The divisor cannot be zero.

     */

    function div(uint256 a, uint256 b) internal pure

        return div(a, b, "SafeMath: division by zero

    }



    /**

     * @dev Returns the integer division of two unsi

     * division by zero. The result is rounded towar
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * `revert` opcode (which leaves remaining gas u
     * uses an invalid opcode to revert (consuming a
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function div(uint256 a, uint256 b, string memory
        require(b > 0, errorMessage);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is n

        return c;
    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * Reverts when dividing by zero.
     *
     * Counterpart to Solidity's `%` operator. This
     * opcode (which leaves remaining gas untouched)
     * invalid opcode to revert (consuming all remai
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function mod(uint256 a, uint256 b) internal pure
        return mod(a, b, "SafeMath: modulo by zero")
    }

    /**
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     *
     * Counterpart to Solidity's `%` operator. This
     * opcode (which leaves remaining gas untouched)
     * invalid opcode to revert (consuming all remai
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function mod(uint256 a, uint256 b, string memory
        require(b != 0, errorMessage);
        return a % b;
    }
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
contract OSP{



    using SafeMath for uint256;



//========================================EVENTS======

    event StakeEvent(address indexed staker, address

    event UnstakeEvent(address indexed unstaker, add

    event RewardEvent(address indexed staker, addres

    event RewardStake(address indexed staker, addres




//========================================STAKING POOL

    address public Axiatoken;

    address public OracleIndexFunds;


```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
bool public stakingEnabled;



uint256 constant private FLOAT_SCALAR = 2**64;

uint256 public MINIMUM_STAKE = 10000000000000000

    uint256  public MIN_DIVIDENDS_DUR = 18 hours



    uint public infocheck;



struct User {

            uint256 balance;

            uint256 frozen;

            int256 scaledPayout;

            uint256 staketime;

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        uint256 totalFrozen;

        mapping(address => User) users;

        uint256 scaledPayoutPerToken;

        address admin;

}



Info private info;





constructor() public {



info.admin = msg.sender;

stakingEnabled = false;

}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    modifier onlyCreator() {

    require(msg.sender == info.admin, "Ownable:

    _;

}



modifier onlyAxiaToken() {

    require(msg.sender == Axiatoken, "Authorizat

    _;

}



     function tokenconfigs(address _axiatoken, a

        require(_axiatoken != _oracleindex, "Ins

        require(_axiatoken != address(0) && _ora

    Axiatoken = _axiatoken;

    OracleIndexFunds = _oracleindex;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function _minStakeAmount(uint256 _number) on

            MINIMUM_STAKE = _number*100000000000

    }

    function stakingStatus(bool _status) public
    require(Axiatoken != address(0) && OracleInd
    stakingEnabled = _status;
}

function MIN_DIVIDENDS_DUR_TIME(uint256 _minDura

    MIN_DIVIDENDS_DUR = _minDuration;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
//======================================USER WRITE==



        function StakeTokens(uint256 _tokens) extern

                _stake(_tokens);

        }



        function UnstakeTokens(uint256 _tokens) exte

                _unstake(_tokens);

        }




//======================================USER READ===



        function totalFrozen() public view returns (

                return info.totalFrozen;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
function frozenOf(address _user) public view ret

            return info.users[_user].frozen;

    }



    function dividendsOf(address _user) public v



        if(info.users[_user].staketime < MIN_DIV

            return 0;

        }else{

         return uint256(int256(info.scaledPayout

        }

    }




    function userData(address _user) public view
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
                return (totalFrozen(), frozenOf(_use

        }

//========================================ACTION CALLS

        function _stake(uint256 _amount) internal {

            require(stakingEnabled, "Staking not yet

                require(IERC20(OracleIndexFunds).bal
                require(frozenOf(msg.sender) + _amou
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        info.users[msg.sender].staketime = n

        info.totalFrozen += _amount;

        info.users[msg.sender].frozen += _am



        info.users[msg.sender].scaledPayout

        IERC20(OracleIndexFunds).transferFro



emit StakeEvent(msg.sender, address(this), _

}










function _unstake(uint256 _amount) internal
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        info.totalFrozen -= _amount;

        info.users[msg.sender].frozen -= _am

        info.users[msg.sender].scaledPayout 



        require(IERC20(OracleIndexFunds).tra

emit UnstakeEvent(address(this), msg.sender,



TakeDividends();



}





function TakeDividends() public returns (uin



        uint256 _dividends = dividendsOf(msg
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
            require(IERC20(Axiatoken).transfer(m

            emit RewardEvent(msg.sender, address


            return _dividends;



    }




function scaledToken(uint _amount) external only


            info.scaledPayoutPerToken += _amount

            infocheck = info.scaledPayoutPerToke

            return true;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
function mulDiv (uint x, uint y, uint z) public

      (uint l, uint h) = fullMul (x, y);

      assert (h < z);

      uint mm = mulmod (x, y, z);

      if (mm > l) h -= 1;

      l -= mm;

      uint pow2 = z & -z;

      z /= pow2;

      l /= pow2;

      l += h * ((-pow2) / pow2 + 1);

      uint r = 1;

      r *= 2 - z * r;

      r *= 2 - z * r;
```

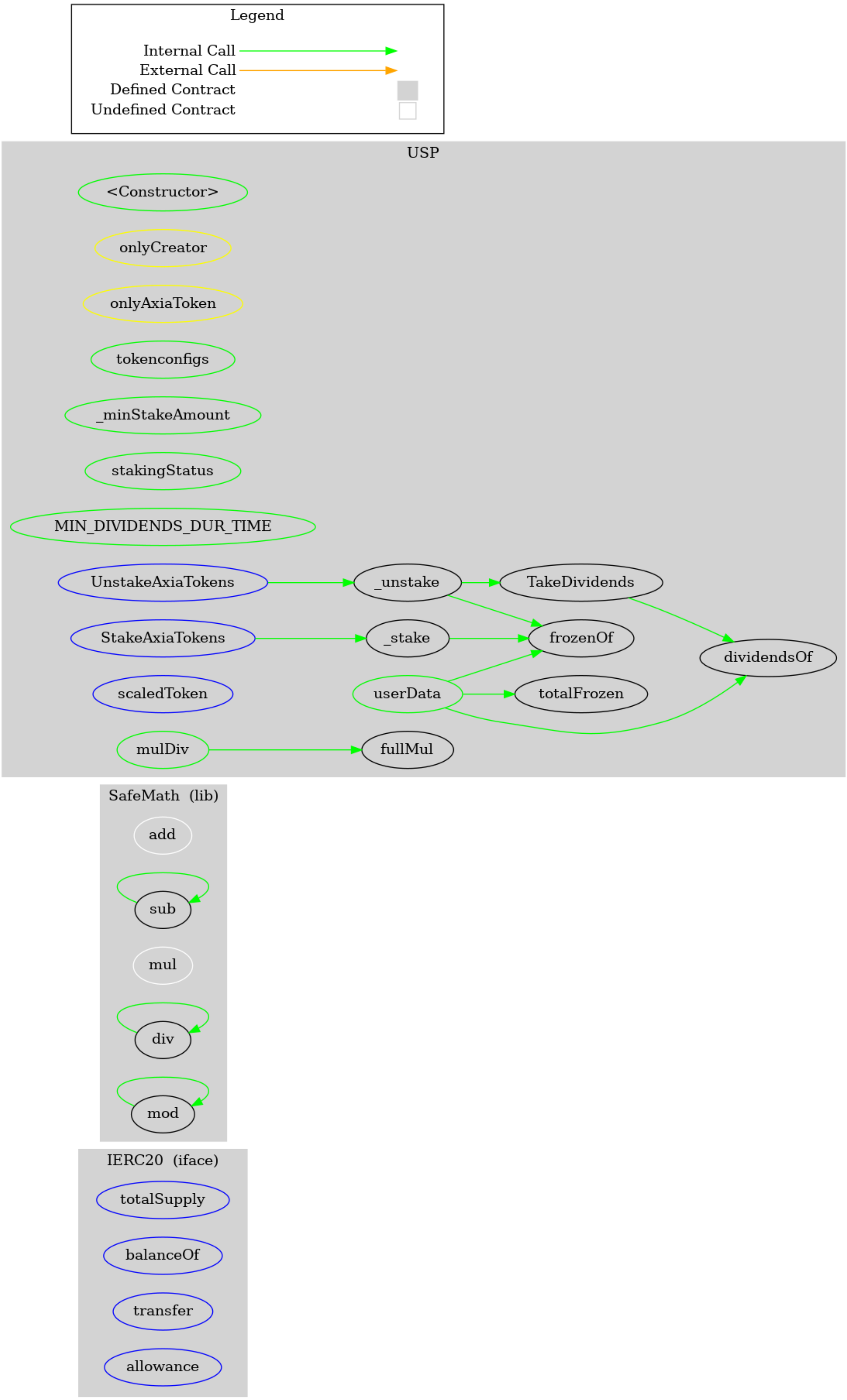
Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        r *= 2 - z * r;

        r *= 2 - z * r;

        r *= 2 - z * r;

        r *= 2 - z * r;

        return l * r;

    }



     function fullMul (uint x, uint y) private pure

        uint mm = mulmod (x, y, uint (-1));

        l = x * y;

        h = mm - l;

        if (mm < l) h -= 1;
```

DETAILS: SWAP FUND

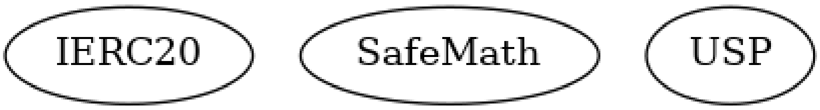
FUNCTION GRAPH

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.



Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

INHERITENCE CHART



FUNCTIONS OVERVIEW

```
($) = payable function
# = non-constant function

Int = Internal
Ext = External
Pub = Public

+ [Int] IERC20
   - [Ext] totalSupply
   - [Ext] balanceOf
   - [Ext] transfer #
   - [Ext] allowance
   - [Ext] approve #
   - [Ext] transferFrom #

+ [Lib] SafeMath
   - [Int] add
   - [Int] sub
   - [Int] sub
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    - [Int] mod

 +  USP
    - [Pub]  #
    - [Pub] tokenconfigs #
       - modifiers: onlyCreator
    - [Pub] _minStakeAmount #
       - modifiers: onlyCreator
    - [Pub] stakingStatus #
       - modifiers: onlyCreator
    - [Pub] MIN_DIVIDENDS_DUR_TIME #
       - modifiers: onlyCreator
    - [Ext] StakeAxiaTokens #
    - [Ext] UnstakeAxiaTokens #
    - [Pub] totalFrozen
    - [Pub] frozenOf
    - [Pub] dividendsOf
    - [Pub] userData
    - [Int] _stake #
    - [Int] _unstake #
    - [Pub] TakeDividends #
    - [Ext] scaledToken #
       - modifiers: onlyAxiaToken
    - [Pub] mulDiv
    - [Prv] fullMul
```

SOURCE CODE

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
/**

 *Submitted for verification at Etherscan.io on 2020

*/



/*



* @dev This is the Axia Protocol Staking pool 3 cont

* a part of the protocol where stakers are rewarded

* when they make stakes of liquidity tokens from the


* stakers reward come from the daily emission from t

* this happens daily and upon the reach of a new epo

* halvings are experienced on the emitting amount of


* on the 11th epoch all the tokens would have been d
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
 * which would now be coming from the accumulated bas

 * stakers are not charged any fee for unstaking.

*/

pragma solidity 0.6.4;

interface IERC20 {

    function totalSupply() external view returns (ui

    function balanceOf(address account) external vie

    function transfer(address recipient, uint256 amo

    function allowance(address owner, address spende
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function approve(address spender, uint256 amount

    function transferFrom(address sender, address re

    event Transfer(address indexed from, address ind

    event Approval(address indexed owner, address in
}

library SafeMath {
    /**
     * @dev Returns the addition of two unsigned int
     * overflow.
     *
     * Counterpart to Solidity's `+` operator.
     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     *
     * - Addition cannot overflow.
     */
    function add(uint256 a, uint256 b) internal pure
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow

        return c;
    }

    /**
     * @dev Returns the subtraction of two unsigned
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     *
     * - Subtraction cannot overflow.
     */
    function sub(uint256 a, uint256 b) internal pure
        return sub(a, b, "SafeMath: subtraction over
    }


    /**
     * @dev Returns the subtraction of two unsigned
     * overflow (when the result is negative).
     *
     * Counterpart to Solidity's `-` operator.
     *
     * Requirements:
     *
     * - Subtraction cannot overflow.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function sub(uint256 a, uint256 b, string memory

        require(b <= a, errorMessage);

        uint256 c = a - b;



        return c;

    }



    /**

     * @dev Returns the multiplication of two unsign

     * overflow.

     *

     * Counterpart to Solidity's `*` operator.

     *

     * Requirements:

     *

     * - Multiplication cannot overflow.
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function mul(uint256 a, uint256 b) internal pure
        // Gas optimization: this is cheaper than re
        // benefit is lost if 'b' is also tested.
        // See: https://github.com/OpenZeppelin/open
        if (a == 0) {
            return 0;
        }

        uint256 c = a * b;
        require(c / a == b, "SafeMath: multiplicatio

        return c;
    }

    /**
     * @dev Returns the integer division of two unsi
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     *
     * Counterpart to Solidity's `/` operator. Note:
     * `revert` opcode (which leaves remaining gas u
     * uses an invalid opcode to revert (consuming a
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function div(uint256 a, uint256 b) internal pure
        return div(a, b, "SafeMath: division by zero
    }


    /**
     * @dev Returns the integer division of two unsi
     * division by zero. The result is rounded towar
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * Counterpart to Solidity's `/` operator. Note:
     * `revert` opcode (which leaves remaining gas u
     * uses an invalid opcode to revert (consuming a
     *
     * Requirements:
     *
     * - The divisor cannot be zero.
     */
    function div(uint256 a, uint256 b, string memory
        require(b > 0, errorMessage);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is n

        return c;
    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
 * @dev Returns the remainder of dividing two un

 * Reverts when dividing by zero.

 *

 * Counterpart to Solidity's `%` operator. This

 * opcode (which leaves remaining gas untouched)

 * invalid opcode to revert (consuming all remai

 *

 * Requirements:

 *

 * - The divisor cannot be zero.

 */

function mod(uint256 a, uint256 b) internal pure

    return mod(a, b, "SafeMath: modulo by zero")

}



/**
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * Reverts with custom message when dividing by

     *

     * Counterpart to Solidity's `%` operator. This

     * opcode (which leaves remaining gas untouched)

     * invalid opcode to revert (consuming all remai

     *

     * Requirements:

     *

     * - The divisor cannot be zero.

     */

    function mod(uint256 a, uint256 b, string memory

        require(b != 0, errorMessage);

        return a % b;

    }

}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
contract USP{


    using SafeMath for uint256;


//======================================EVENTS======

    event StakeEvent(address indexed staker, address

    event UnstakeEvent(address indexed unstaker, add

    event RewardEvent(address indexed staker, addres

    event RewardStake(address indexed staker, addres




//======================================STAKING POOL

    address public Axiatoken;

    address public UniswapV2;

```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    uint256 constant private FLOAT_SCALAR = 2**64;

    uint256 public MINIMUM_STAKE = 1000000000000000

        uint256 public MIN_DIVIDENDS_DUR = 18 hours;



        uint public infocheck;



    struct User {

                uint256 balance;

                uint256 frozen;

                int256 scaledPayout;

                uint256 staketime;

        }



        struct Info {

                uint256 totalSupply;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
                mapping(address => User) users;

                uint256 scaledPayoutPerToken; //pool

                address admin;

        }



        Info private info;




        constructor() public {



        info.admin = msg.sender;

        stakingEnabled = false;

        }



//=======================================ADMINSTRATIC
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(msg.sender == info.admin, "Ownable:

        _;

    }



    modifier onlyAxiaToken() {

        require(msg.sender == Axiatoken, "Authorizat

        _;

    }



         function tokenconfigs(address _axiatoken, a

        require(_axiatoken != _univ2, "Insertion of

        require(_axiatoken != address(0) && _univ2 !

        Axiatoken = _axiatoken;

        UniswapV2 = _univ2;

        return true;

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        function _minStakeAmount(uint256 _number) on

                MINIMUM_STAKE = _number*100000000000

        }

        function stakingStatus(bool _status) public
        require(Axiatoken != address(0) && UniswapV2
        stakingEnabled = _status;
    }

    function MIN_DIVIDENDS_DUR_TIME(uint256 _minDura

        MIN_DIVIDENDS_DUR = _minDuration;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
//======================================USER WRITE==

        function StakeAxiaTokens(uint256 _tokens) ex
                _stake(_tokens);
        }

        function UnstakeAxiaTokens(uint256 _tokens)
                _unstake(_tokens);
        }



//======================================USER READ===

        function totalFrozen() public view returns (
                return info.totalFrozen;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
function frozenOf(address _user) public view ret

            return info.users[_user].frozen;

    }



    function dividendsOf(address _user) public v



        if(info.users[_user].staketime < MIN_DIV

            return 0;

        }else{

         return uint256(int256(info.scaledPayout

       }

    }




    function userData(address _user) public view
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        uint256 userDividends, uint256 userStaketime

                return (totalFrozen(), frozenOf(_use

        }

//========================================ACTION CALLS

        function _stake(uint256 _amount) internal {

            require(stakingEnabled, "Staking not yet

                require(IERC20(UniswapV2).balanceOf(
                require(frozenOf(msg.sender) + _amou
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
            info.users[msg.sender].staketime = n
            info.totalFrozen += _amount;
            info.users[msg.sender].frozen += _am

            info.users[msg.sender].scaledPayout
            IERC20(UniswapV2).transferFrom(msg.s

    emit StakeEvent(msg.sender, address(this), _
    }




    function _unstake(uint256 _amount) internal
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        info.totalFrozen -= _amount;

        info.users[msg.sender].frozen -= _am

        info.users[msg.sender].scaledPayout


        require(IERC20(UniswapV2).transfer(m

emit UnstakeEvent(address(this), msg.sender,


TakeDividends();


}




function TakeDividends() public returns (uin
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
            require(_dividends >= 0, "you do not

            info.users[msg.sender].scaledPayout

            require(IERC20(Axiatoken).transfer(m

            emit RewardEvent(msg.sender, address


            return _dividends;




    }




function scaledToken(uint _amount) external only


            info.scaledPayoutPerToken += _amount

            infocheck = info.scaledPayoutPerToke

            return true;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
}

function mulDiv (uint x, uint y, uint z) public

    (uint l, uint h) = fullMul (x, y);

    assert (h < z);

    uint mm = mulmod (x, y, z);

    if (mm > l) h -= 1;

    l -= mm;

    uint pow2 = z & -z;

    z /= pow2;

    l /= pow2;

    l += h * ((-pow2) / pow2 + 1);

    uint r = 1;

    r *= 2 - z * r;

    r *= 2 - z * r;
```

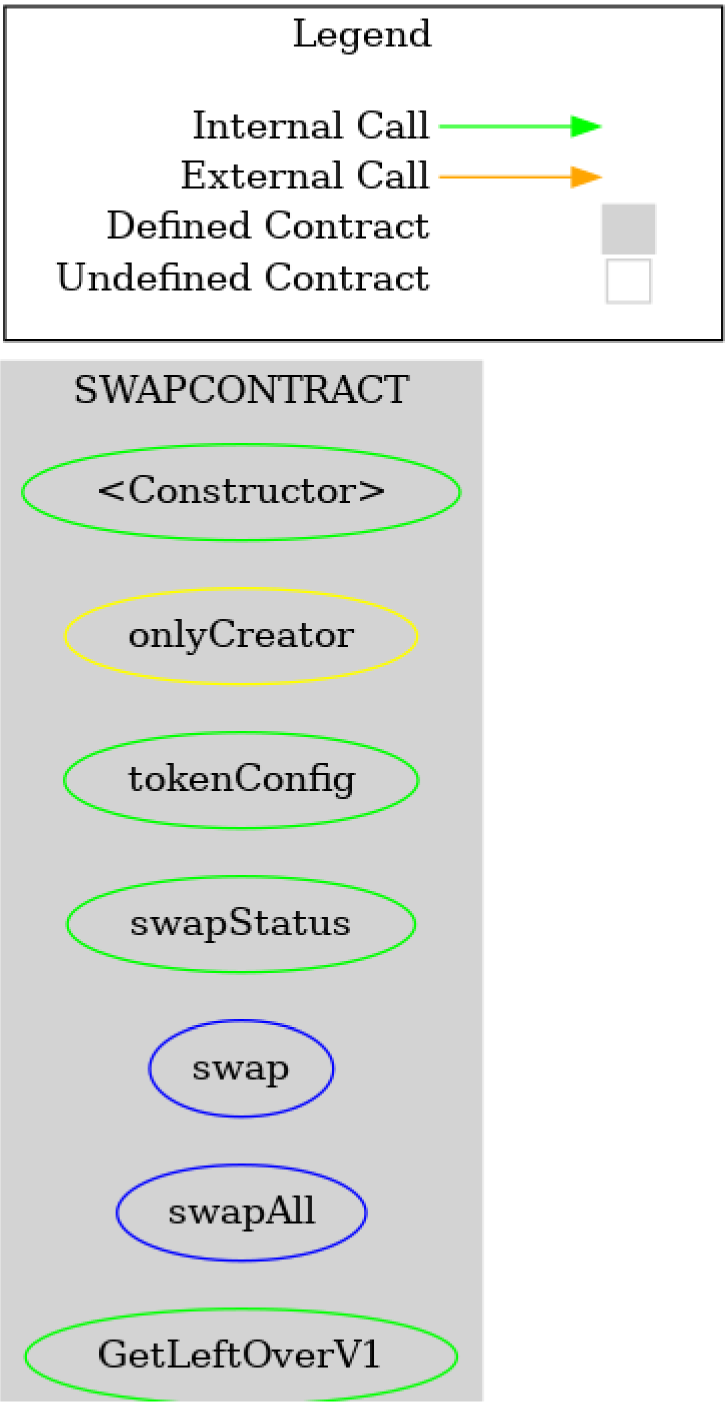
Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
      r *= 2 - z * r;

      r *= 2 - z * r;

      r *= 2 - z * r;

      r *= 2 - z * r;

      r *= 2 - z * r;

      return l * r;

}



 function fullMul (uint x, uint y) private pure

      uint mm = mulmod (x, y, uint (-1));

      l = x * y;

      h = mm - l;

      if (mm < l) h -= 1;

}
```

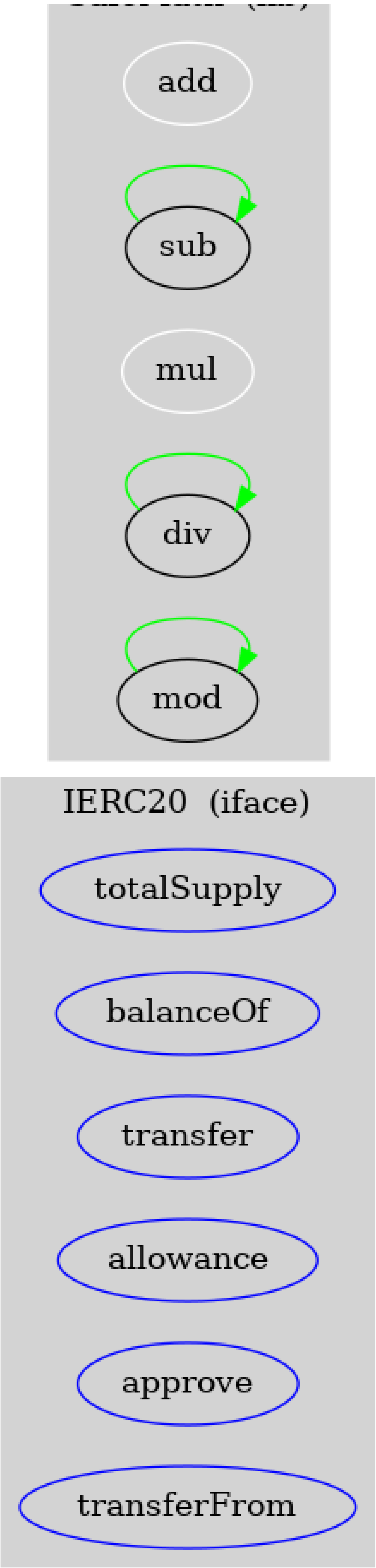
Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

DETAILS: TOKEN SWAP CONTRACT

FUNCTION GRAPH



Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.



Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

FUNCTIONS OVERVIEW

```
($) = payable function
# = non-constant function

Int = Internal
Ext = External
Pub = Public

+ [Int] IERC20
   - [Ext] totalSupply
   - [Ext] balanceOf
   - [Ext] transfer #
   - [Ext] allowance
   - [Ext] approve #
   - [Ext] transferFrom #

+ [Lib] SafeMath
   - [Int] add
   - [Int] sub
   - [Int] sub
   - [Int] mul
   - [Int] div
   - [Int] div
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
 +  SWAPCONTRACT
    - [Pub]  #
    - [Pub] tokenConfig #
       - modifiers: onlyCreator
    - [Pub] swapStatus #
       - modifiers: onlyCreator
    - [Ext] swap #
    - [Ext] swapAll #
    - [Pub] GetLeftOverV1 #
       - modifiers: onlyCreator
    - [Pub] GetLeftOverV2 #
       - modifiers: onlyCreator
```

SOURCE CODE

Click here to download the source code as a .sol file.

```
pragma solidity 0.6.4;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function totalSupply() external view returns (ui

    function balanceOf(address account) external vie

    function transfer(address recipient, uint256 amo

    function allowance(address owner, address spende

    function approve(address spender, uint256 amount

    function transferFrom(address sender, address re

    event Transfer(address indexed from, address ind

    event Approval(address indexed owner, address in
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
library SafeMath {
    /**
     * @dev Returns the addition of two unsigned int
     * overflow.
     *
     * Counterpart to Solidity's `+` operator.
     *
     * Requirements:
     *
     * - Addition cannot overflow.
     */
    function add(uint256 a, uint256 b) internal pure
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        return c;

    }



    /**

     * @dev Returns the subtraction of two unsigned

     * overflow (when the result is negative).

     *

     * Counterpart to Solidity's `-` operator.

     *

     * Requirements:

     *

     * - Subtraction cannot overflow.

     */

    function sub(uint256 a, uint256 b) internal pure

        return sub(a, b, "SafeMath: subtraction over

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
/**
 * @dev Returns the subtraction of two unsigned
 * overflow (when the result is negative).
 *
 * Counterpart to Solidity's `-` operator.
 *
 * Requirements:
 *
 * - Subtraction cannot overflow.
 */
function sub(uint256 a, uint256 b, string memory
    require(b <= a, errorMessage);
    uint256 c = a - b;

    return c;
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    /**
     * @dev Returns the multiplication of two unsign
     * overflow.
     *
     * Counterpart to Solidity's `*` operator.
     *
     * Requirements:
     *
     * - Multiplication cannot overflow.
     */
    function mul(uint256 a, uint256 b) internal pure
        // Gas optimization: this is cheaper than re
        // benefit is lost if 'b' is also tested.
        // See: https://github.com/OpenZeppelin/open
        if (a == 0) {
            return 0;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
            uint256 c = a * b;

            require(c / a == b, "SafeMath: multiplicatio


            return c;

        }



        /**

         * @dev Returns the integer division of two unsi

         * division by zero. The result is rounded towar

         *

         * Counterpart to Solidity's `/` operator. Note:

         * `revert` opcode (which leaves remaining gas u

         * uses an invalid opcode to revert (consuming a

         *

         * Requirements:
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * - The divisor cannot be zero.

     */

    function div(uint256 a, uint256 b) internal pure

        return div(a, b, "SafeMath: division by zero

    }



    /**

     * @dev Returns the integer division of two unsi

     * division by zero. The result is rounded towar

     *

     * Counterpart to Solidity's `/` operator. Note:

     * `revert` opcode (which leaves remaining gas u

     * uses an invalid opcode to revert (consuming a

     *

     * Requirements:

     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     */
    function div(uint256 a, uint256 b, string memory
        require(b > 0, errorMessage);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is n

        return c;
    }


    /**
     * @dev Returns the remainder of dividing two un
     * Reverts when dividing by zero.
     *
     * Counterpart to Solidity's `%` operator. This 
     * opcode (which leaves remaining gas untouched)
     * invalid opcode to revert (consuming all remai
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     * Requirements:

     *

     * - The divisor cannot be zero.

     */

    function mod(uint256 a, uint256 b) internal pure

        return mod(a, b, "SafeMath: modulo by zero")

    }



    /**

     * @dev Returns the remainder of dividing two un

     * Reverts with custom message when dividing by

     *

     * Counterpart to Solidity's `%` operator. This

     * opcode (which leaves remaining gas untouched)

     * invalid opcode to revert (consuming all remai

     *
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
     *

     * - The divisor cannot be zero.

     */

    function mod(uint256 a, uint256 b, string memory

        require(b != 0, errorMessage);

        return a % b;

    }

}







contract SWAPCONTRACT{



   using SafeMath for uint256;



   address public V1;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
   bool swapEnabled;

   address administrator;



   constructor() public {



            administrator = msg.sender;

                swapEnabled = false;



        }



//======================================ADMINSTRATIC



        modifier onlyCreator() {

        require(msg.sender == administrator, "Ownabl

        _;

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function tokenConfig(address _v1Address, address 

        require(_v1Address != address(0) && _v2Addres

        V1 = _v1Address;

        V2 = _v2Address;

        return true;



    }





    function swapStatus(bool _status) public onlyCrea

        require(V1 != address(0) && V2 != address(0),

        swapEnabled = _status;

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function swap(uint256 _amount) external returns(b



        require(swapEnabled, "Swap not yet initialize

        require(_amount > 0, "Invalid amount to swap"

        require(IERC20(V1).balanceOf(msg.sender) >= _

        require(IERC20(V2).balanceOf(address(this)) >

        require(IERC20(V1).allowance(msg.sender, addr



        require(IERC20(V1).transferFrom(msg.sender, a

        require(IERC20(V2).transfer(msg.sender, _amou



        return true;



    }



    function swapAll() external returns(bool){
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

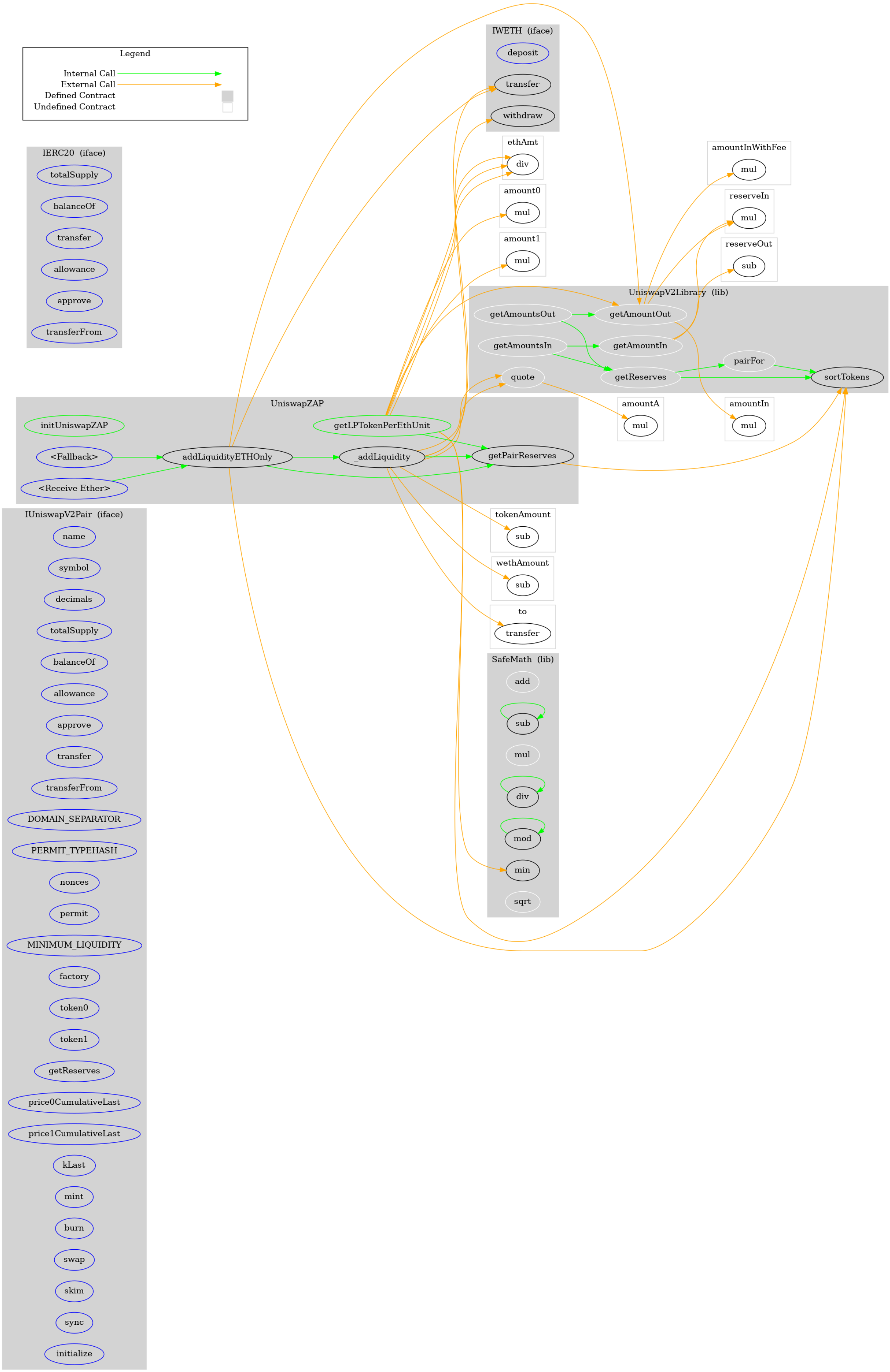
```
        require(swapEnabled, "Swap not yet initialize

        uint v1userbalance = IERC20(V1).balanceOf(msg

        uint v2contractbalance = IERC20(V2).balanceOf



        require(v1userbalance > 0, "You cannot swap o

        require(v2contractbalance >= v1userbalance, "

        require(IERC20(V1).allowance(msg.sender, addr



        require(IERC20(V1).transferFrom(msg.sender, a

        require(IERC20(V2).transfer(msg.sender, v1use



        return true;



    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(administrator != address(0));

        require(administrator != address(this));

        require(V1 != address(0) && V2 != address(0),

        uint bal = IERC20(V1).balanceOf(address(this))

        require(IERC20(V1).transfer(administrator, bal



    }



    function GetLeftOverV2() public onlyCreator return



        require(administrator != address(0));

        require(administrator != address(this));

        require(V1 != address(0) && V2 != address(0),

        uint bal = IERC20(V2).balanceOf(address(this))
```

DETAILS: UNISWAP ZAP

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.



Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

FUNCTIONS OVERVIEW

```
($) = payable function
# = non-constant function

Int = Internal
Ext = External
Pub = Public

 + [Int] IUniswapV2Pair
   - [Ext] name
   - [Ext] symbol
   - [Ext] decimals
   - [Ext] totalSupply
   - [Ext] balanceOf
   - [Ext] allowance
   - [Ext] approve #
   - [Ext] transfer #
   - [Ext] transferFrom #
   - [Ext] DOMAIN_SEPARATOR
   - [Ext] PERMIT_TYPEHASH
   - [Ext] nonces
   - [Ext] permit #
   - [Ext] MINIMUM_LIQUIDITY
   - [Ext] factory
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    - [Ext] price0CumulativeLast
    - [Ext] price1CumulativeLast
    - [Ext] kLast
    - [Ext] mint #
    - [Ext] burn #
    - [Ext] swap #
    - [Ext] skim #
    - [Ext] sync #
    - [Ext] initialize #

 + [Int] IWETH
    - [Ext] deposit ($)
    - [Ext] transfer #
    - [Ext] withdraw #

 + [Int] IERC20
    - [Ext] totalSupply
    - [Ext] balanceOf
    - [Ext] transfer #
    - [Ext] allowance
    - [Ext] approve #
    - [Ext] transferFrom #

 + [Lib] SafeMath
    - [Int] add
    - [Int] sub
    - [Int] sub
    - [Int] mul
    - [Int] div
    - [Int] div
    - [Int] mod
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
 + [Lib] UniswapV2Library
    - [Int] sortTokens
    - [Int] pairFor
    - [Int] getReserves
    - [Int] quote
    - [Int] getAmountOut
    - [Int] getAmountIn
    - [Int] getAmountsOut
    - [Int] getAmountsIn

 +  UniswapZAP
    - [Pub] initUniswapZAP #
    - [Ext]  ($)
    - [Ext]  ($)
    - [Pub] addLiquidityETHOnly ($)
    - [Int] _addLiquidity #
    - [Pub] getLPTokenPerEthUnit
    - [Int] getPairReserves
```

SOURCE CODE

Click here to download the source code as a .sol file.

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
interface IUniswapV2Pair {
    event Approval(address indexed owner, address in
    event Transfer(address indexed from, address ind

    function name() external pure returns (string me
    function symbol() external pure returns (string 
    function decimals() external pure returns (uint8
    function totalSupply() external view returns (ui
    function balanceOf(address owner) external view 
    function allowance(address owner, address spende

    function approve(address spender, uint value) ex
    function transfer(address to, uint value) extern
    function transferFrom(address from, address to, 
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function DOMAIN_SEPARATOR() external view return
    function PERMIT_TYPEHASH() external pure returns
    function nonces(address owner) external view ret

    function permit(address owner, address spender,

    event Mint(address indexed sender, uint amount0,
    event Burn(address indexed sender, uint amount0,
    event Swap(
        address indexed sender,
        uint amount0In,
        uint amount1In,
        uint amount0Out,
        uint amount1Out,
        address indexed to
    );
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function MINIMUM_LIQUIDITY() external pure retur
    function factory() external view returns (addres
    function token0() external view returns (address
    function token1() external view returns (address
    function getReserves() external view returns (ui
    function price0CumulativeLast() external view re
    function price1CumulativeLast() external view re
    function kLast() external view returns (uint);

    function mint(address to) external returns (uint
    function burn(address to) external returns (uint
    function swap(uint amount0Out, uint amount1Out,
    function skim(address to) external;
    function sync() external;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
}


interface IWETH {
    function deposit() external payable;
    function transfer(address to, uint value) extern
    function withdraw(uint) external;
}


/**
 * @dev Interface of the ERC20 standard as defined i
 */
interface IERC20 {
    function totalSupply() external view returns (ui
    function balanceOf(address account) external vie
    function transfer(address recipient, uint256 amo
    function allowance(address owner, address spende
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    function transferFrom(address sender, address re

    event Transfer(address indexed from, address ind

    event Approval(address indexed owner, address in

}




/**

 * @dev Wrappers over Solidity's arithmetic operatio

 * checks.

 */

library SafeMath {


    function add(uint256 a, uint256 b) internal pure

        uint256 c = a + b;

        require(c >= a, "SafeMath: addition overflow
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        return c;
    }


    function sub(uint256 a, uint256 b) internal pure
        return sub(a, b, "SafeMath: subtraction over
    }


    function sub(uint256 a, uint256 b, string memory
        require(b <= a, errorMessage);
        uint256 c = a - b;

        return c;
    }


    function mul(uint256 a, uint256 b) internal pure
        // Gas optimization: this is cheaper than re
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        // See: https://github.com/OpenZeppelin/open
        if (a == 0) {
            return 0;
        }

        uint256 c = a * b;
        require(c / a == b, "SafeMath: multiplicatio

        return c;
    }

    function div(uint256 a, uint256 b) internal pure
        return div(a, b, "SafeMath: division by zero
    }

    function div(uint256 a, uint256 b, string memory
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is n

        return c;
    }

    function mod(uint256 a, uint256 b) internal pure
        return mod(a, b, "SafeMath: modulo by zero")
    }

    function mod(uint256 a, uint256 b, string memory
        require(b != 0, errorMessage);
        return a % b;
    }

    function min(uint x, uint y) internal pure retur
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    }

    // babylonian method (https://en.wikipedia.org/w
    function sqrt(uint y) internal pure returns (uin
        if (y > 3) {
            z = y;
            uint x = y / 2 + 1;
            while (x < z) {
                z = x;
                x = (y / x + x) / 2;
            }
        } else if (y != 0) {
            z = 1;
        }
    }
}
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
library UniswapV2Library {

    using SafeMath for uint;


    // returns sorted token addresses, used to handl

    function sortTokens(address tokenA, address toke

        require(tokenA != tokenB, 'UniswapV2Library:

        (token0, token1) = tokenA < tokenB ? (tokenA

        require(token0 != address(0), 'UniswapV2Libr

    }


    // calculates the CREATE2 address for a pair wit

    function pairFor(address factory, address tokenA

        (address token0, address token1) = sortToken
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
                    hex'ff',

                    factory,

                    keccak256(abi.encodePacked(token0, t

                    hex'96e8ac4277198ff8b6f785478aa9a39f

                ))));

        }



        // fetches and sorts the reserves for a pair

        function getReserves(address factory, address to

            (address token0,) = sortTokens(tokenA, token

            (uint reserve0, uint reserve1,) = IUniswapV2

            (reserveA, reserveB) = tokenA == token0 ? (r

        }



        // given some amount of an asset and pair reserv

        function quote(uint amountA, uint reserveA, uint
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(reserveA > 0 && reserveB > 0, 'Unisw

        amountB = amountA.mul(reserveB) / reserveA;

    }



    // given an input amount of an asset and pair re

    function getAmountOut(uint amountIn, uint reserv

        require(amountIn > 0, 'UniswapV2Library: INS

        require(reserveIn > 0 && reserveOut > 0, 'Un

        uint amountInWithFee = amountIn.mul(997);

        uint numerator = amountInWithFee.mul(reserve

        uint denominator = reserveIn.mul(1000).add(a

        amountOut = numerator / denominator;

    }



    // given an output amount of an asset and pair r

    function getAmountIn(uint amountOut, uint reserv
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        require(reserveIn > 0 && reserveOut > 0, 'Un
        uint numerator = reserveIn.mul(amountOut).mu
        uint denominator = reserveOut.sub(amountOut)
        amountIn = (numerator / denominator).add(1);
    }

    // performs chained getAmountOut calculations on
    function getAmountsOut(address factory, uint amo
        require(path.length >= 2, 'UniswapV2Library:
        amounts = new uint[](path.length);
        amounts[0] = amountIn;
        for (uint i; i < path.length - 1; i++) {
            (uint reserveIn, uint reserveOut) = getR
            amounts[i + 1] = getAmountOut(amounts[i]
        }
    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    // performs chained getAmountIn calculations on
    function getAmountsIn(address factory, uint amou
        require(path.length >= 2, 'UniswapV2Library:
        amounts = new uint[](path.length);
        amounts[amounts.length - 1] = amountOut;
        for (uint i = path.length - 1; i > 0; i--) {
            (uint reserveIn, uint reserveOut) = getR
            amounts[i - 1] = getAmountIn(amounts[i],
        }
    }
}

contract UniswapZAP {

    using SafeMath for uint256;
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    address public _tokenWETHPair;
    IWETH public _WETH;
    bool private initialized;


    function initUniswapZAP(address token, address W
        require(!initialized);
        _token = token;
        _WETH = IWETH(WETH);
        _tokenWETHPair = tokenWethPair;
        initialized = true;
    }


    fallback() external payable {
        if(msg.sender != address(_WETH)){
             addLiquidityETHOnly(msg.sender);
        }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
receive() external payable {

    if(msg.sender != address(_WETH)){

        addLiquidityETHOnly(msg.sender);

    }

}




function addLiquidityETHOnly(address payable to)

    require(to != address(0), "Invalid address")



    uint256 buyAmount = msg.value.div(2);

    require(buyAmount > 0, "Insufficient ETH amo

    _WETH.deposit{value : msg.value}();



     (uint256 reserveWeth, uint256 reserveTokens)

    uint256 outTokens = UniswapV2Library.getAmou
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        _WETH.transfer(_tokenWETHPair, buyAmount);



        (address token0, address token1) = UniswapV2

        IUniswapV2Pair(_tokenWETHPair).swap(_token =



        _addLiquidity(outTokens, buyAmount, to);



    }




    function _addLiquidity(uint256 tokenAmount, uint

        (uint256 wethReserve, uint256 tokenReserve)



        uint256 optimalTokenAmount = UniswapV2Librar



        uint256 optimalWETHAmount;

        if (optimalTokenAmount > tokenAmount) {
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
    optimalTokenAmount = tokenAmount;

}

else

    optimalWETHAmount = wethAmount;



assert(_WETH.transfer(_tokenWETHPair, optima

assert(IERC20(_token).transfer(_tokenWETHPai



IUniswapV2Pair(_tokenWETHPair).mint(to);



//refund dust

if (tokenAmount > optimalTokenAmount)

    IERC20(_token).transfer(to, tokenAmount.



if (wethAmount > optimalWETHAmount) {

    uint256 withdrawAmount = wethAmount.sub(
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
            to.transfer(withdrawAmount);

        }

    }




    function getLPTokenPerEthUnit(uint ethAmt) publi

        (uint256 reserveWeth, uint256 reserveTokens)

        uint256 outTokens = UniswapV2Library.getAmou

        uint _totalSupply =  IUniswapV2Pair(_tokenWE


        (address token0, ) = UniswapV2Library.sortTo

        (uint256 amount0, uint256 amount1) = token0

        (uint256 _reserve0, uint256 _reserve1) = tok

        liquidity = SafeMath.min(amount0.mul(_totalS

    }
```

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

```
        (address token0,) = UniswapV2Library.sortTok
        (uint256 reserve0, uint reserve1,) = IUniswa
        (wethReserves, tokenReserves) = token0 == _t
    }


}
```

PRINT EXPANDED SECTIONS

GO HOME

Copyright 2021 © Solidity Finance LLC. All rights reserved. Please review our Terms & Conditions, Privacy Policy, and other legal information here.

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.

Please review our Terms & Conditions, Privacy Policy, and other legal information here. By using this site, you explicitly agree to these terms.