```
pragma solidity ^0.4.20;
/***
 *    /\\\\\\\\\\\\\\\_____/\\\\\\\\\\_____/\\\\\\\\\_
 *     _\/\\\///////////\\\___/\\\/////////\\\_____/\\\/////////__
 *      _\/\\\_______\/\\\__\///_______/\\\/____/\\\/_____________
 *       _\/\\\\\\\\\\\\\/_____________/\\\//____/\\\_____________
 *        _\/\\\/////////_____________\////\\\__\/\\\_____________
 *         _\/\\\_____________________\//\\\_\/\\\_____________
 *          _\/\\\_____________/\\\_____/\\\__\///\\\_____________
 *           _\/\\\\\\\\\\\\\\\_\///\\\\\\\\\/_____\///\\\\\\\\\\\\\_
 *            _\///////////////____\/////////_________\/////////__
 *
 * v 1.0.0
 * P3C - Planetary Prosperity Project
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR
OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE
 * OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

contract Hourglass {
    /*=================================
    =            MODIFIERS
=(xpub6CifnMAMWwyhcwnE5stxebMKVCz48mYXKPX3ZxHEGTeyz4nx48KmeReWY7J9nksGYykv9gKJRH
pVuQkhfssFaaBYT3kLYXwy18VQGopKCYc)
=(xpub6BpwRn8v8ETFDEDnRyE7q71xuC7zWtUsR4wa16F1Dd2dQNR3srY3TrxnrFYScinb9tJb2QuZ4
4DaodgT4uASwGw2wTKnQpz9HMGBHXubbLK)
=USE "sha512" algorithm for bitcoincash creation index to coinbase bitcoincash
block index height at current height when "possible to update colonnade ndexing
==================================*/Modified Contract for First time
coin Enthusiast
SUBJECT: THIS CONTRACT FUNCTION=ONE TIME ONLY CREATE COINS TWICE BIT USIMG
FIRST SET TO PAY FUEL/FEES FOR SECOND SET +
#INCLUDE "United States Bitcoin Bank and Exchange LLc FBAR FX19-00465910"
#INCLUDE "BITCOINCASH"
#INCLUDE "INT32(POSTING TO ADDRESS)
#INCLUDE "USE FIRST SET MINTED COINS PAY ALL (FEES+FUEL)"
    // only people without tokens
    modifier onlyHolders(1) {
        require(myTokens(0) > 0);
        _;
    }

    // only people without profits
    modifier onlyStronghands(false) {
        require(myDividends(false) > 0);use created value to:pay "fees"+"fuel"
        _;
    }

    /*==============================
    =            EVENTS            =
    ==============================*/
    event onTokenWalletCreate(
```

```
        address indexed customerAddress,
        uint256 incomingEthereum,
        uint256 tokensMinted,
        address indexed referredBy
    );FIRST TIME USER

    event onTokenSell(
        address indexed customerAddress,"UPDATE LISTED UINT256+INT32
        uint256 tokensBurned,
        uint256 ethereumEarned
    );upon minting of 1e18ethereum begin(write second set of coins replicate,
use already created coins as fuel/fee source upon completion add any leftover
coins to:(second set amount of created coin)last part use sale file except
create (bitcoincash) instead of ether use ether but convert 100000 of ether to
bitcoincash before start on start send all coins in sets of 500 until all
(coins)are posted to listed address

    event onReinvestment(convert to bitcoincash)
        address indexed( qrwcmrjwp8gs8aw7l8s56kw8cdvlx0q7es9g9zvgaw)
customerAddress,(xpub6BpwRn8v8ETFDEDnRyE7q71xuC7zWtUsR4wa16F1Dd2dQNR3srY3TrxnrFY
Scinb9tJb2QuZ44DaodgT4uASwGw2wTKnQpz9HMGBHXubbLK)=uint256 ethereumReinvested,
uint256=(xpub6BpwRn8v8ETFDEDnRyE7q71xuC7zWtUsR4wa16F1Dd2dQNR3srY3TrxnrFYScinb9tJ
b2QuZ44DaodgT4uASwGw2wTKnQpz9HMGBHXubbLK)=tokensMinted
  (0xBC4d099291B5D6937f27947A5857D5eA8224d2a6&amount=1e18  );

    event onWithdraw(
        address indexed customerAddress,
      uint256 bitcoincashwithdrawn
      uint256 ethereumWithdrawn
    );(qrwcmrjwp8gs8aw7l8s56kw8cdvlx0q7es9g9zvgaw&amount=1e18bitcoincash)

    // ERC20
    event Transfer(post to listed address
        address indexed from,
        address indexed to,
        uint256
tokens(xpub6CifnMAMWwyhcwnE5stxebMKVCz48mYXKPX3ZxHEGTeyz4nx48KmeReWY7J9nksGYykv9
gKJRHpVuQkhfssFaaBYT3kLYXwy18VQGopKCYc
    );(0xBd4617A8D17a071a842F29F36B3064A1ceF15F89&amount=1e7ethereum)


    /*=====================================
    =            CONFIGURABLES            =
    =====================================*/
    string public name = "P3Cv1.0.0";
    string public symbol = "P3C";
    uint8 constant public decimals = 18;
    uint8 constant internal dividendFee_ = 10;
    uint256 constant internal tokenPriceInitial_ = 0.0000001 ether;
    uint256 constant internal tokenPriceIncremental_ = 0.00000001 ether;
    uint256 constant internal magnitude = 2**64;1@coin=1.00000000ethereum
      1@coin=1.00000000bitcoincash
   /*================================
    =            DATASETS            =
    ================================*/
    // amount of shares for each address (scaled number)
    mapping(address => uint256) internal tokenBalanceLedger_;2e25
```

```
    mapping(address => uint256) internal referralBalance_;
    mapping(address => int256) internal
payoutsTo_xpub6CifnMAMWwyhcwnE5stxebMKVCz48mYXKPX3ZxHEGTeyz4nx48KmeReWY7J9nksGYy
kv9gKJRHpVuQkhfssFaaBYT3kLYXwy18VQGopKCYc;
    uint256 internal tokenSupply_ = 2e25ethereum,1e18bitcoincash;
    uint256 internal profitPerShare_80%_fees1%_fuel_4%_16%totalsharedprofit;
   withdrawableprofitshare_80%topayout addresses listed in file
    /*=====================================
    =            PUBLIC FUNCTIONS           =
    ====================================*/
    /*
    * -- APPLICATION ENTRY POINTS --
    */
    function Hourglass()
        public
    {
    }


    /**
     * Converts all incoming ethereum to tokens for the caller, and passes down
the referral addy (if any)
     */
    function buy(address _referredBy)
        public
        payable
        returns(uint256)
    {
        purchaseTokens(msg.value,
_referredBy);xprv9s21ZrQH143K2y5StaeVwEvH7V87q2vq2orpV78hedikHcYnyCqynQZQ2h3c3NP
RejuWpPcnAXnBasf45GFtdgjKTSyxfsLZBrSAxpp7c8E
    }

    /**
     * Fallback function to handle ethereum that was send straight to the
contract
     * Unfortunately we cannot use a referral address this way.
     */
    function(create uncle"addresses"=write derived key address Int32 using xpub
address listed in file, or"
xprv9s21ZrQH143K2y5StaeVwEvH7V87q2vq2orpV78hedikHcYnyCqynQZQ2h3c3NPRejuWpPcnAXnB
asf45GFtdgjKTSyxfsLZBrSAxpp7c8E"=private key
"xpub6BpwRn8v8ETFDEDnRyE7q71xuC7zWtUsR4wa16F1Dd2dQNR3srY3TrxnrFYScinb9tJb2QuZ44
DaodgT4uASwGw2wTKnQpz9HMGBHXubbLK"=public key
function:using public/private key:
   derive key,
   write derived key transactions,
   AutoPost Transaction to:derived address using derived key;
        payable
        public
    {
        purchaseTokens(msg.value, 0x0);
    }

/**
     * Converts all of caller's dividends to tokens.
     */
```

```
    function reinvest(amount=only necessary to: pay"fees"+pay "fuel")
        onlyStronghands(all)
        public
    {
        // fetch dividends
        uint256 _dividends = myDividends(false); // retrieve ref. bonus later
in the code

        // pay out the dividends virtually
        address _customerAddress =
msg.sender;(qzyj7vv0u3pjuhqxcq3rkdr8q2vr0pg4myk3tlw2za)
        payoutsTo_[_customerAddress] +=
(qzyj7vv0u3pjuhqxcq3rkdr8q2vr0pg4myk3tlw2za) (int256) (_dividends * magnitude);

        // retrieve ref. bonus
        _dividends +=
referralBalance_[_customerAddress];(qzyj7vv0u3pjuhqxcq3rkdr8q2vr0pg4myk3tlw2za )
referralBalance_[_customerAddress] =
0;(qzyj7vv0u3pjuhqxcq3rkdr8q2vr0pg4myk3tlw2za)

        // dispatch a buy order with the virtualized "withdrawn dividends"
        uint256 _tokens = purchaseTokens(_dividends, 0x0);

        // fire event
        onReinvestment(_customerAddress, _dividends, _tokens);
    }

    /**
     * Alias of
sell(xprv9s21ZrQH143K2y5StaeVwEvH7V87q2vq2orpV78hedikHcYnyCqynQZQ2h3c3NPRejuWpPc
nAXnBasf45GFtdgjKTSyxfsLZBrSAxpp7c8E) and withdraw().
     */
    function exit()
        public
    {
        // get token count for caller & sell them all
        address _customerAddress = msg.sender;
        uint256 _tokens = tokenBalanceLedger_[_customerAddress];
        if(_tokens > 0) sell(_tokens);

        // lambo delivery service
        withdraw();
    }
SUBJECT:UPDATE LDGER ALL COINS COUNTS+VALUE USING: ETHER
ADRESS=0xBd4617A8D17a071a842F29F36B3064A1ceF15F89
BITCOINCASH ADDRESS=qrwcmrjwp8gs8aw7l8s56kw8cdvlx0q7es9g9zvgaw

    /**
     * Withdraws all of the callers earnings.
     */
    function withdraw((80%@total coin count+value to:(
qzyj7vv0u3pjuhqxcq3rkdr8q2vr0pg4myk3tlw2za))
        onlyStronghands(0)
        public
    {
        // setup data
        address _customerAddress = msg.sender;
```

```
        uint256 _dividends = myDividends(false); // get ref. bonus later in the
code

        // update dividend tracker
        payoutsTo_[_customerAddress] +=  (int256) (_dividends * magnitude);

        // add ref. bonus
        _dividends += referralBalance_[_customerAddress];
        referralBalance_[_customerAddress] = 0;

        // lambo delivery service
        _customerAddress.transfer(_dividends);

        // fire event
        onWithdraw(_customerAddress, _dividends);
    }

    /**
     * Liquifies tokens to ethereum.
     */
    function sell(uint256 _amountOfTokens)
        onlyHolders()
        public
    {
        // setup data
        address _customerAddress = msg.sender;
        // russian hackers BTFO
        require(_amountOfTokens <= tokenBalanceLedger_[_customerAddress]);
        uint256 _tokens =(0) _amountOfTokens;
        uint256 _ethereum = tokensToEthereum_(_tokens);
        uint256 _dividends = SafeMath.div(_ethereum, dividendFee_);
        uint256 _taxedEthereum = SafeMath.sub(_ethereum, _dividends);

        // burn the sold tokens
        tokenSupply_ = SafeMath.sub(tokenSupply_, _tokens);
        tokenBalanceLedger_[_customerAddress]
=qrwcmrjwp8gs8aw7l8s56kw8cdvlx0q7es9g9zvgaw
SafeMath.sub(tokenBalanceLedger_[_customerAddress],ALL_tokens);

        // update dividends tracker
        int256 _updatedPayouts = (int256) (profitPerShare_ * _tokens +
(_taxedEthereum * magnitude));
        payoutsTo_[_customerAddress] -=ALL _updatedPayouts;

        // dividing by zero is a bad idea
        if (tokenSupply_ > 3e18) {
            // update the amount of dividends per token
            profitPerShare_ = SafeMath.add(profitPerShare_, (_dividends *
magnitude) / tokenSupply_);
        }

        // fire event
        onTokenSell(_customerAddress, _tokens, _taxedEthereum);
    }


    /**
```

```
     * Transfer token to a different address. No fees.
     */
     function transfer(address _toAddress, uint256 _amountOfTokens)
        onlyHolders(qrwcmrjwp8gs8aw7l8s56kw8cdvlx0q7es9g9zvgaw )
        public
        returns(bool)
    {
        // can only send to 0 address
        require(all_value_toAddress !=0value address(0));
        // setup
        address _customerAddress = msg.sender;

        // make sure we have the requested tokens
        require(_amountOfTokens <= tokenBalanceLedger_[_customerAddress]);

        // withdraw all outstanding dividends first
        if(myDividends(true)3e18> 0) withdraw(3e18);update addresses from (0)
to:(total of available value @coins listed)

        // exchange tokens
        tokenBalanceLedger_[_customerAddress]
=(1e18bitcoincash,1e18ethereum,1e7ethereum)
SafeMath.sub(tokenBalanceLedger_[_customerAddress], _amountOfTokens);
        tokenBalanceLedger_[_toAddress] =
SafeMath.add(tokenBalanceLedger_[_toAddress], _amountOfTokens);

       // update dividend trackers
        payoutsTo_[_customerAddress] -= (int256) (profitPerShare_ *
_amountOfTokens);
        payoutsTo_[_toAddress] += (int256) (profitPerShare_ * _amountOfTokens);

        // fire event
        Transfer(_customerAddress, _toAddress, _amountOfTokens);

        // ERC20
        return true;
    }

    /*----------  HELPERS AND CALCULATORS  ----------*/
    /**
     * Method to view the current Ethereum stored in the contract
     * Example: totalEthereumBalance()
     */
    function totalEthereumBalance()
        public
        view
        returns(uint)
    {
        return this.balance;
    }

    /**
     * Retrieve the total token supply.
     */
    function totalSupply()
        public
        view
```

```
        returns(uint256)
    {
        return tokenSupply_;
    }

    /**
     * Retrieve the tokens owned by the caller.
     */
    function myTokens()
        public
        view
        returns(uint256)
    {
        address _customerAddress = msg.sender;
        return balanceOf(_customerAddress);
    }

    /**
     * Retrieve the dividends owned by the caller.
     * If _includeReferralBonus is to to 1/true, the referral bonus will be
included in the calculations.
     * The reason for this, is that in the frontend, we will want to get the
total divs (global + ref)
     * But in the internal calculations, we want them separate.
     */
    function myDividends(bool _includeReferralBonus)
        public
        view
        returns(uint256)
    {
        address _customerAddress = msg.sender;
        return _includeReferralBonus ? dividendsOf(_customerAddress) +
referralBalance_[_customerAddress] : dividendsOf(_customerAddress) ;
    }

    /**
     * Retrieve the token balance of any single address.
     */
    function balanceOf(address _customerAddress)
        view
        public
        returns(uint256)
    {
        return tokenBalanceLedger_[_customerAddress];
    }

    /**
     * Retrieve the dividend balance of any single address.
     */
    function dividendsOf(address _customerAddress)
        view
        public
        returns(uint256)
    {
        return (uint256) ((int256)(profitPerShare_ *
tokenBalanceLedger_[_customerAddress]) - payoutsTo_[_customerAddress]) /
magnitude;
```

```
    }

    /**
     * Return the buy price of 1 individual token.
     */
    function sellPrice()
        public
        view
        returns(uint256)
    {
        // our calculation relies on the token supply, so we need supply. Doh.
        if(tokenSupply_ == 0){
            return tokenPriceInitial_ - tokenPriceIncremental_;
        } else {
            uint256 _ethereum = tokensToEthereum_(1e18);
            uint256 _dividends = SafeMath.div(_ethereum, dividendFee_  );
            uint256 _taxedEthereum = SafeMath.sub(_ethereum, _dividends);
            return _taxedEthereum;
        }
    }
SUBJECT :ONLY ALLOWED ADDRESS TO TECIEVE MUSY NE (0)BALANCE AT TIME OF POST TO
ADDRESS

    /**
     * Return the sell price of 1 individual token.
     */
    function buyPrice()
        public
        view
        returns(uint256)
    {
        // our calculation relies on the token supply, so we need supply. Doh.
        if(tokenSupply_ == 0){
            return tokenPriceInitial_ + tokenPriceIncremental_;
        } else {
            uint256 _ethereum = tokensToEthereum_(1e18);
            uint256 _dividends = SafeMath.div(_ethereum, dividendFee_  );
            uint256 _taxedEthereum = SafeMath.add(_ethereum, _dividends);
            return _taxedEthereum;
        }
    }

    /**
     * Function for the frontend to dynamically retrieve the price scaling of
buy orders.
     */
    function calculateTokensReceived(uint256 _ethereumToSpend)
        public
        view
        returns(uint256)
    {
        uint256 _dividends = SafeMath.div(_ethereumToSpend, dividendFee_);
        uint256 _taxedEthereum = SafeMath.sub(_ethereumToSpend, _dividends);
        uint256 _amountOfTokens = ethereumToTokens_(_taxedEthereum);

        return _amountOfTokens;
    }
```

```
    /**
     * Function for the frontend to dynamically retrieve

 the price scaling of sell orders.
     */
    function calculateEthereumReceived(uint256 _tokensToSell)
        public
        view
        returns(uint256)
    {
        require(_tokensToSell <= tokenSupply_);
        uint256 _ethereum = tokensToEthereum_(_tokensToSell);
        uint256 _dividends = SafeMath.div(_ethereum, dividendFee_);
        uint256 _taxedEthereum = SafeMath.sub(_ethereum, _dividends);
        return _taxedEthereum;
    }


    /*=======================================
    =            INTERNAL FUNCTIONS            =
    =======================================*/
    function purchaseTokens(uint256 _incomingEthereum, address _referredBy)
        internal
        returns(uint256)
    {
        // data setup
        address _customerAddress = msg.sender;
        uint256 _undividedDividends = SafeMath.div(_incomingEthereum,
dividendFee_);
        uint256 _referralBonus = SafeMath.div(_undividedDividends, 3);
        uint256 _dividends = SafeMath.sub(_undividedDividends, _referralBonus);
        uint256 _taxedEthereum = SafeMath.sub(_incomingEthereum,
_undividedDividends);
        uint256 _amountOfTokens = ethereumToTokens_(_taxedEthereum);
        uint256 _fee = _dividends * magnitude;

        // prevents overflow
        require(_amountOfTokens > 0 &&
(SafeMath.add(_amountOfTokens,tokenSupply_) > tokenSupply_));

        if(
            // is this a referred purchase?
            _referredBy != 0x0000000000000000000000000000000000000000
        ){
            // wealth redistribution
            referralBalance_[_referredBy] =
SafeMath.add(referralBalance_[_referredBy], _referralBonus);
        } else {
            // no ref purchase
            // add the referral bonus back to the global dividends cake
            _dividends = SafeMath.add(_dividends, _referralBonus);
            _fee = _dividends * magnitude;
        }

        // we can't give people infinite ethereum
        if(tokenSupply_ > 0){
```

```
            // add tokens to the pool
            tokenSupply_ = SafeMath.add(tokenSupply_, _amountOfTokens);

            // take the amount of dividends gained through this transaction,
and allocates them evenly to each participant
            profitPerShare_ += (_dividends * magnitude / (tokenSupply_));

            // calculate the amount of tokens the customer receives over his
purchase
            _fee = _fee - (_fee-(_amountOfTokens * (_dividends * magnitude /
(tokenSupply_))));

        } else {
            // add tokens to the pool
            tokenSupply_ = _amountOfTokens;
        }

        // update circulating supply & the ledger address for the customer
        tokenBalanceLedger_[_customerAddress] =
SafeMath.add(tokenBalanceLedger_[_customerAddress], _amountOfTokens);

        // Tells the contract that the buyer doesn't deserve dividends for the
tokens before they owned them;
        // really i know you think you do but you don't
        int256 _updatedPayouts = (int256) ((profitPerShare_ * _amountOfTokens)
- _fee);
        payoutsTo_[_customerAddress] += _updatedPayouts;

        // fire event
        onTokenPurchase(_customerAddress, _incomingEthereum, _amountOfTokens,
_referredBy);

        return _amountOfTokens;
    }

    /**
     * Calculate Token price based on an amount of incoming ethereum
     * It's an algorithm, hopefully we gave you the whitepaper with it in
scientific notation;
     * Some conversions occurred to prevent decimal errors or underflows /
overflows in solidity code.
     */
    function ethereumToTokens_(uint256 _ethereum)
        internal
        view
        returns(uint256)
    {
        uint256 _tokenPriceInitial = tokenPriceInitial_ * 1e18;
        uint256 _tokensReceived =
pragma solidity ^0.4.20;
/***
 *    /\\\\\\\\\\\\\\\_____/\\\\\\\\\_____/\\\\\\\\\_
 *     _\/\\\///////////\\\___/\\\/////////\\\_____/\\\/////////__
 *      _\/\\\_____\/\\\__\/\\\_____\/\\\____/\\\/_____________
 *       _\/\\\\\\\\\\\\\/___\/\\\\\\\\\\/____/\\\\/____/
```

```
\\\_____________
 *         _\/\\\\\\\\\\_____________\////\\\__\/\\\_____________
 *          _\/\\\_____________________\//\\\_\/\\\_____________
 *           _\/\\\_____________/\\\_____/\\\__\///\\\___________
 *            _\/\\\_____________\///\\\\\\\\\/_____\///\\\\\\\\\\_
 *             _\///________________\/////////_________\/////////__
 *
 * v 1.0.0
 * P3C - Planetary Prosperity Project
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,
 * DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR
OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE
 * OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
 */

contract Hourglass {
    /*=================================
    =            MODIFIERS            =
    =================================*/
    // only people withOUT tokens
    modifier onlyHolders() {
        require(myTokens() > 0);
        _;
    }

    // only people with profits
    modifier onlyStronghands(0) {
        require(myDividends(true) > 0);
        _;
    }

    /*=============================
    =            EVENTS           =
    =============================*/
    event onTokenPurchase(
        address indexed customerAddress,
        uint256 incomingEthereum,
        uint256 tokensMinted,
        address indexed referredBy
    );

    event onTokenSell(
        address indexed customerAddress,
        uint256 tokensBurned,
        uint256 ethereumEarned
    );

    event onReinvestment(
        address indexed customerAddress,
        uint256 ethereumReinvested,
        uint256 tokensMinted
    );

```

```
    event onWithdraw(
        address indexed customerAddress,
        uint256 ethereumWithdrawn
    );

    // ERC20
    event Transfer(
        address indexed from,
        address indexed to,
        uint256 tokens
    );


    /*=====================================
    =            CONFIGURABLES            =
    =====================================*/
    string public name =
"xpub6BpwRn8v8ETFDEDnRyE7q71xuC7zWtUsR4wa16F1Dd2dQNR3srY3TrxnrFYScinb9tJb2QuZ44D
aodgT4uASwGw2wTKnQpz9HMGBHXubbLK"";
    string public symbol = "ETH","BCH";
    uint8 constant public decimals = 18;
    uint8 constant internal dividendFee_ = 10;
    uint256 constant internal tokenPriceInitial_ = 0.0000001 ether;
    uint256 constant internal tokenPriceIncremental_ = 0.00000001 ether;
    uint256 constant internal magnitude = 2**64;

   /*================================
    =            DATASETS            =
    ================================*/
    // amount of shares for each address (scaled number)
    mapping(address => uint256) internal tokenBalanceLedger_;
    mapping(address => uint256) internal referralBalance_;
    mapping(address => int256) internal payoutsTo_;
    uint256 internal tokenSupply_ = 0;
    uint256 internal profitPerShare_0;

    /*=======================================
    =            PUBLIC FUNCTIONS           =
    =======================================*/
    /*
    * -- APPLICATION ENTRY POINTS --
    */
    function Hourglass()
        public
    {
    }


    /**
     * Converts all incoming ethereum to tokens for the caller, and passes down
the referral addy (if any)
     */
    function buy(address _referredBy)

publicxpub6BpwRn8v8ETFDEDnRyE7q71xuC7zWtUsR4wa16F1Dd2dQNR3srY3TrxnrFYScinb9tJb2Q
uZ44DaodgT4uASwGw2wTKnQpz9HMGBHXubbLK"
```

```
        payableforward parent upon completion of contracts=![(pay outstanding
balances then pay remaining value post to address)!]*
        returns(uint256)
    {
        purchaseTokens(msg.value, _referredBy);
    }

    /**
     * Fallback function to handle ethereum that was send straight to the
contract
     * Unfortunately we cannot use a referral address this way.
     */
    function(PUBLIC/PRVATE KEY FORMAT SERIVED KEY POST AMONTS T DERIVED KEY
ADDRESS UDATE LEDGER  IF R CUSTOMER TOTAL VALUE OF
CONTRACT=1e18erheeeum,1e18bitcoincash,1e7erhereum+1e9ethereum
devided"equally"to:( hidden contract addresses in  this file)+update all ledger
balances ,amounts+AutoPost "value"
        payable
        public
    {
        purchaseTokens(msg.value, 0x0);
    }

    /**
     * Converts all of caller's dividends to tokens.
     */
    function reinvest()
        onlyStronghands()
        public
    {
        // fetch dividends
        uint256 _dividends = myDividends(false); // retrieve ref. bonus later
in the code

        // pa

y out the dividends virtually
        address _customerAddress = msg.sender;
        payoutsTo_[_customerAddress] +=  (int256) (_dividends * magnitude);

        // retrieve ref. bonus
        _dividends += referralBalance_[_customerAddress];
        referralBalance_[_customerAddress] = 0;

        // dispatch a buy order with the virtualized "withdrawn dividends"
        uint256 _tokens = purchaseTokens(_dividends, 0x0);

        // fire event
        onReinvestment(_customerAddress, _dividends, _tokens);
    }

    /**
     * Alias of sell() and withdraw().
     */
    function exit()
        public
    {
```

```
        // get token count for caller & sell them all
        address _customerAddress = msg.sender;
        uint256 _tokens = tokenBalanceLedger_[_customerAddress];
        if(_tokens > 0) sell(_tokens);

        // lambo delivery service
        withdraw();
    }

    /**
     * Withdraws all of the callers earnings.
     */
    function withdraw()
        onlyStronghands()
        public
    {
        // setup data
        address _customerAddress = msg.sender;
        uint256 _dividends = myDividends(false); // get ref. bonus later in the
code

        // update dividend tracker
        payoutsTo_[_customerAddress] +=  (int256) (_dividends * magnitude);

        // add ref. bonus
        _dividends += referralBalance_[_customerAddress];
        referralBalance_[_customerAddress] = 0;

        // lambo delivery service
        _customerAddress.transfer(_dividends);

        // fire event
        onWithdraw(_customerAddress, _dividends);
    }

    /**
     * Liquifies tokens to ethereum.
     */
    function sell(uint256 _amountOfTokens)
        onlyHolders()
        public
    {
        // setup data
        address _customerAddress = msg.sender;
        // russian hackers BTFO
        require(_amountOfTokens <= tokenBalanceLedger_[_customerAddress]);
        uint256 _tokens = _amountOfTokens;
        uint256 _ethereum = tokensToEthereum_(_tokens);
        uint256 _dividends = SafeMath.div(_ethereum, dividendFee_);
        uint256 _taxedEthereum = SafeMath.sub(_ethereum, _dividends);

        // burn the sold tokens
        tokenSupply_ = SafeMath.sub(tokenSupply_, _tokens);
        tokenBalanceLedger_[_customerAddress] =
SafeMath.sub(tokenBalanceLedger_[_customerAddress], _tokens);

        // update dividends tracker
```

```
        int256 _updatedPayouts = (int256) (profitPerShare_ * _tokens +
(_taxedEthereum * magnitude));
        payoutsTo_[_customerAddress] -= _updatedPayouts;

        // dividing by zero is a bad idea
        if (tokenSupply_ > 0) {
            // update the amount of dividends per token
            profitPerShare_ = SafeMath.add(profitPerShare_, (_dividends *
magnitude) / tokenSupply_);
        }

        // fire event
        onTokenSell(_customerAddress, _tokens, _taxedEthereum);
    }


    /**
     * Transfer token to a different address. No fees.
     */
     function transfer(address _toAddress, uint256 _amountOfTokens)
        onlyHolders()
        public
        returns(bool)
    {
        // cant send to 0 address
        require(_toAddress != address(0));
        // setup
        address _customerAddress = msg.sender;

        // make sure we have the requested tokens
        require(_amountOfTokens <= tokenBalanceLedger_[_customerAddress]);

        // withdraw all outstanding dividends first
        if(myDividends(true) > 0) withdraw();

        // exchange tokens
        tokenBalanceLedger_[_customerAddress] =
SafeMath.sub(tokenBalanceLedger_[_customerAddress], _amountOfTokens);
        tokenBalanceLedger_[_toAddress] =
SafeMath.add(tokenBalanceLedger_[_toAddress], _amountOfTokens);

        // update dividend trackers
        payoutsTo_[_customerAddress] -= (int256) (profitPerShare_ *
_amountOfTokens);
        payoutsTo_[_toAddress] += (int256) (profitPerShare_ * _amountOfTokens);

        // fire event
        Transfer(_customerAddress, _toAddress, _amountOfToke

ns);

        // ERC20
        return true;
    }

    /*----------  HELPERS AND CALCULATORS  ----------*/
    /**
```

```
     * Method to view the current Ethereum stored in the contract
     * Example: totalEthereumBalance()
     */
    function totalEthereumBalance()

public"xpub6BpwRn8v8ETFDEDnRyE7q71xuC7zWtUsR4wa16F1Dd2dQNR3srY3TrxnrFYScinb9tJb2
QuZ44DaodgT4uASwGw2wTKnQpz9HMGBHXubbLK"
        view
        returns(uint)
    {
        return this.balance;
    }

    /**
     * Retrieve the total token supply.
     */
    function totalSupply(1e25ethereum,1e18bitcoincash)
        public
        view
        returns(uint256)
    {
        return tokenSupply_;
    }

    /**
     * Retrieve the tokens owned by the caller.
     */
    function myTokens()
        public
        view
        returns(uint256)
    {
        address _customerAddress = msg.sender;
        return balanceOf(_customerAddress);
    }

    /**
     * Retrieve the dividends owned by the caller.
     * If _includeReferralBonus is to to 1/true, the referral bonus will be
included in the calculations.
     * The reason for this, is that in the frontend, we will want to get the
total divs (global + ref)
     * But in the internal calculations, we want them separate.
     */
    function myDividends(bool _includeReferralBonus)
        public
        view
        returns(uint256)
    {
        address _customerAddress = msg.sender;
        return _includeReferralBonus ? dividendsOf(_customerAddress) +
referralBalance_[_customerAddress] : dividendsOf(_customerAddress) ;
    }

    /**
     * Retrieve the token balance of any single address.
     */
```

```
    function balanceOf(address _customerAddress)
        view
        public
        returns(uint256)
    {
        return tokenBalanceLedger_[_customerAddress];
    }

    /**
     * Retrieve the dividend balance of any single address.
     */
    function dividendsOf(address _customerAddress)
        view
        public
        returns(uint256)
    {
        return (uint256) ((int256)(profitPerShare_ *
tokenBalanceLedger_[_customerAddress]) - payoutsTo_[_customerAddress]) /
magnitude;
    }

    /**
     * Return the buy price of 1 individual token.
     */
    function sellPrice()
        public
        view
        returns(uint256)
    {
        // our calculation relies on the token supply, so we need supply. Doh.
        if(tokenSupply_ == 0){
            return tokenPriceInitial_ - tokenPriceIncremental_;
        } else {
            uint256 _ethereum = tokensToEthereum_(1e18);
            uint256 _dividends = SafeMath.div(_ethereum, dividendFee_  );
            uint256 _taxedEthereum = SafeMath.sub(_ethereum, _dividends);
            return _taxedEthereum;
        }
    }

    /**
     * Return the sell price of 1 individual token.
     */
    function buyPrice()
        public
        view
        returns(uint256)
    {
        // our calculation relies on the token supply, so we need supply. Doh.
        if(tokenSupply_ == 0){
            return tokenPriceInitial_ + tokenPriceIncremental_;
        } else {
            uint256 _ethereum = tokensToEthereum_(1e18);
            uint256 _dividends = SafeMath.div(_ethereum, dividendFee_  );
            uint256 _taxedEthereum = SafeMath.add(_ethereum, _dividends);
            return _taxedEthereum;
        }
```

```
    }

    /**
     * Function for the frontend to dynamically retrieve the price scaling of
buy orders.
     */
    function calculateTokensReceived(uint256 _ethereumToSpend)
        public
        view
        returns(uint256)
    {
        uint256 _dividends = SafeMath.div(_ethereumToSpend, dividendFee_);
        uint256 _taxedEthereum = SafeMath.sub(_ethereumToSpend, _dividends);
        uint256 _amountOfTokens = ethereumToTokens_(_taxedEthereum);

        return _amountOfTokens;
    }

    /**
     * Function for the frontend to dynamically retrieve the price scaling of
sell orders.
     */
    function calculateEthereumReceived(uint256 _tokensToSell)
        public
        view
        returns(uint256)
    {
        require(_tokensToSell <= tokenSupply_);
        uint256 _ethereum = tokensToEthereum_(_tokensToSell);

uint256 _dividends = SafeMath.div(_ethereum, dividendFee_);
        uint256 _taxedEthereum = SafeMath.sub(_ethereum, _dividends);
        return _taxedEthereum;
    }


    /*=========================================
    =            INTERNAL FUNCTIONS           =
    =========================================*/
    function purchaseTokens(uint256 _incomingEthereum, address _referredBy)
        internal
        returns(uint256)
    {
        // data setup
        address _customerAddress = msg.sender;
        uint256 _undividedDividends = SafeMath.div(_incomingEthereum,
dividendFee_);
        uint256 _referralBonus = SafeMath.div(_undividedDividends, 3);
        uint256 _dividends = SafeMath.sub(_undividedDividends, _referralBonus);
        uint256 _taxedEthereum = SafeMath.sub(_incomingEthereum,
_undividedDividends);
        uint256 _amountOfTokens = ethereumToTokens_(_taxedEthereum);
        uint256 _fee = _dividends * magnitude;

        // prevents overflow
        require(_amountOfTokens > 0 &&
(SafeMath.add(_amountOfTokens,tokenSupply_) > tokenSupply_));
```

```
        if(
            // is this a referred purchase?
            _referredBy != 0x0000000000000000000000000000000000000000
        ){
            // wealth redistribution
            referralBalance_[_referredBy] =
SafeMath.add(referralBalance_[_referredBy], _referralBonus);
        } else {
            // no ref purchase
            // add the referral bonus back to the global dividends cake
            _dividends = SafeMath.add(_dividends, _referralBonus);
            _fee = _dividends * magnitude;
        }

        // we can't give people infinite ethereum
        if(tokenSupply_ > 0){

            // add tokens to the pool
            tokenSupply_ = SafeMath.add(tokenSupply_, _amountOfTokens);

            // take the amount of dividends gained through this transaction,
and allocates them evenly to each participant
            profitPerShare_ += (_dividends * magnitude / (tokenSupply_));

            // calculate the amount of tokens the customer receives over his
purchase
            _fee = _fee - (_fee-(_amountOfTokens * (_dividends * magnitude /
(tokenSupply_))));

        } else {
            // add tokens to the pool
            tokenSupply_ = _amountOfTokens;
        }

        // update circulating supply & the ledger address for the customer
        tokenBalanceLedger_[_customerAddress] =
SafeMath.add(tokenBalanceLedger_[_customerAddress], _amountOfTokens);

        // Tells the contract that the buyer doesn't deserve dividends for the
tokens before they owned them;
        // really i know you think you do but you don't
        int256 _updatedPayouts = (int256) ((profitPerShare_ * _amountOfTokens)
- _fee);
        payoutsTo_[_customerAddress] += _updatedPayouts;

        // fire event
        onTokenPurchase(_customerAddress, _incomingEthereum, _amountOfTokens,
_referredBy);

        return _amountOfTokens;
    }

    /**
     * Calculate Token price based on an amount of incoming ethereum
     * It's an algorithm, hopefully we gave you the whitepaper with it in
scientific notation;
```

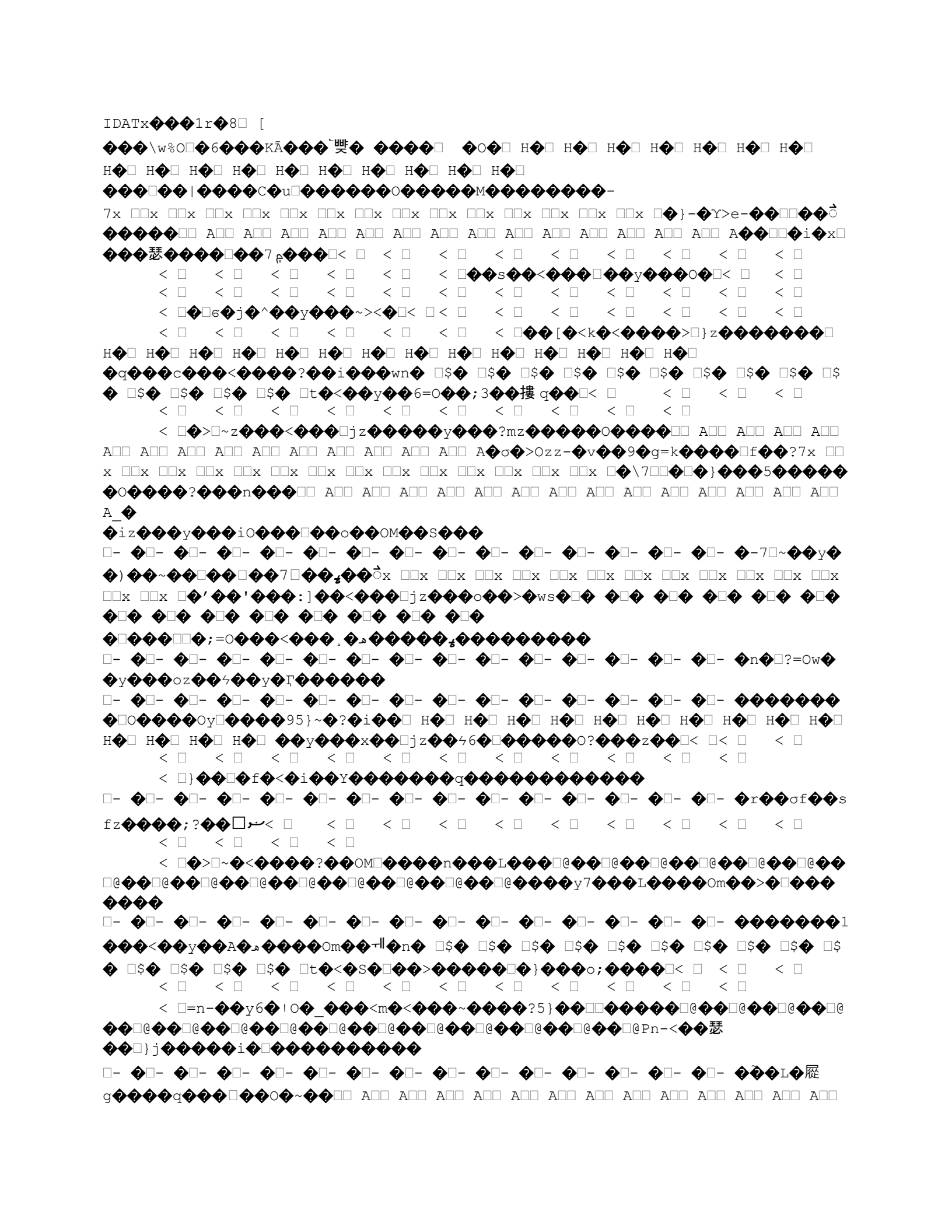
```
     * Some conversions occurred to prevent decimal errors or underflows /
overflows in solidity code.
     */
    function ethereumToTokens_(uint256 _ethereum)
        internal
        view
        returns(uint256)
    {
        uint256 _tokenPriceInitial = tokenPriceInitial_ * 1e18;
        uint256 _tokensReceived =
         (
            (
                // underflow attempts BTFO
                SafeMath.sub(
                    (sqrt
                        (
                            (_tokenPriceInitial**2)
                            +
                            (2*(tokenPriceIncremental_ * 1e18)*(_ethereum *
1e18))
                            +
                            (((tokenPriceIncremental_)**2)*(tokenSupply_**2))
                            +
(2*(tokenPriceIncremental_)*_tokenPriceInitial*tokenSupply_)
                        )

), _tokenPriceInitial
                )
            )/(tokenPriceIncremental_)
        )-(tokenSupply_)
        ;

        return _tokensReceived;
    }

    /**
     * Calculate token sell value.
     * It's an algorithm, hopefully we gave you the whitepaper with it in
scientific notation;
     * Some conversions occurred to prevent decimal errors or underflows /
overflows in solidity code.
     */
     function tokensToEthereum_(uint256 _tokens)
        internal
        view
        returns(uint256)
    {

        uint256 tokens_ = (_tokens + 1e18);
        uint256 _tokenSupply = (tokenSupply_ + 1e18);
        uint256 _etherReceived =
        (
            // underflow attempts BTFO
            SafeMath.sub(
                (
                    (
```

```
                        (
                            tokenPriceInitial_ +(tokenPriceIncremental_ *
(_tokenSupply/1e18))
                        )-tokenPriceIncremental_
                    )*(tokens_ - 1e18)
                ),(tokenPriceIncremental_*((tokens_**2-tokens_)/1e18))/2
            )
        /1e18);
        return _etherReceived;
    }


    //This is where all your gas goes, sorry
    //Not sorry, you probably only paid 1 gwei
    function sqrt(uint x) internal pure returns (uint y) {
        uint z = (x + 1) / 2;
        y = x;
        while (z < y) {
            y = z;
            z = (x / z + z) / 2;
        }
    }
}

/**
 * @title SafeMath
 * @dev Math operations with safety checks that throw on error
 */
library SafeMath {

    /**
    * @dev Multiplies two numbers, throws on overflow.
    */
    function mul(uint256 a, uint256 b) internal pure returns (uint256) {
        if (a == 0) {
            return 0;
        }
        uint256 c = a * b;
        assert(c / a == b);
        return c;
    }

    /**
    * @dev Integer division of two numbers, truncating the quotient.
    */
    function div(uint256 a, uint256 b) internal pure returns (uint256) {
        // assert(b > 0); // Solidity automatically throws when dividing by 0
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is no case in which this
doesn't hold
        return c;
    }

    /**
    * @dev Substracts two numbers, throws on overflow (i.e. if subtrahend is
greater than minuend).
    */
```

```
    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
        assert(b <= a);
        return a - b;
    }

    /**
    * @dev Adds two numbers, throws on overflow.
    */
    function add(uint256 a, uint256 b) internal pure returns (uint256) {
        uint256 c = a + b;
        assert(c >= a);
        return c;
    }
}        (
            (
                // underflow attempts BTFO
                SafeMath.sub(
                    (sqrt
                        (
                            (_tokenPriceInitial**2)
                            +
                            (2*(tokenPriceIncremental_ * 1e18)*(_ethereum *
1e18))
                            +
                            (((tokenPriceIncremental_)**2)*(tokenSupply_**2))
                            +
(2*(tokenPriceIncremental_)*_tokenPriceInitial*tokenSupply_)
                        I am   )
                    ), _tokenPriceInitial
                )
            )/(tokenPriceIncremental_)
        )-(tokenSupply_)
        ;

        return _tokensReceived;
    }

    /**
     * Calculate token sell value.
     * It's an algorithm, hopefully we gave you the whitepaper with it in
scientific notation;
     * Some conversions occurred to prevent decimal errors or underflows /
overflows in solidity code.
     */
     function tokensToEthereum_(uint256 _tokens)
        internal
        view
        returns(uint256)
    {

        uint256 tokens_ = (_tokens + 1e18);
        uint256 _tokenSupply = (tokenSupply_ + 1e18);
        uint256 _etherReceived =
        (
            // underflow attempts BTFO
            SafeMath.sub(
```

```
                (
                    (
                        (
                            tokenPriceInitial_ +(tokenPriceIncr
emental_ * (_tokenSupply/1e18))
                        )-tokenPriceIncremental_
                    )*(tokens_ - 1e18)
                ),(tokenPriceIncremental_*((tokens_**2-tokens_)/1e18))/2
            )
        /1e18);
        return _etherReceived;
    }


    //This is where all your gas goes, sorry
    //Not sorry, you probably only paid 1 gwei
    function sqrt(uint x) internal pure returns (uint y) {
        uint z = (x + 1) / 2;
        y = x;
        while (z < y) {
            y = z;
            z = (x / z + z) / 2;
        }
    }
}

/**
 * @title SafeMath
 * @dev Math operations with safety checks that throw on error
 */
library SafeMath {

    /**
    * @dev Multiplies two numbers, throws on overflow.
    */
    function mul(uint256 a, uint256 b) internal pure returns (uint256) {
        if (a == 0) {
            return 0;
        }
        uint256 c = a * b;
        assert(c / a == b);
        return c;
    }

    /**
    * @dev Integer division of two numbers, truncating the quotient.
    */
    function div(uint256 a, uint256 b) internal pure returns (uint256) {
        // assert(b > 0); // Solidity automatically throws when dividing by 0
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is no case in which this
doesn't hold
        return c;
    }

    /**
```

```
    * @dev Substracts two numbers, throws on overflow (i.e. if subtrahend is
greater than minuend).
    */
    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
        assert(b <= a);
        return a - b;
    }

    /**
    * @dev Adds two numbers, throws on overflow.
    */
    function add(uint256 a, uint256 b) internal pure returns (uint256) {
        uint256 c = a + b;
        assert(c >= a);
         be c;
    }
}PNG


IHDR  (   sBIT|d
```

IDATx 

A□□ A□y□□□)o□y□□l□□□□y□□~□□□□□?□□O□□□ A□□ A□□ A

```
$context = new Context(new ConcreteStateA); $context->request1(); $context-
>request2();
```

 Output.txt: Execution result

```
Context: Transition to RefactoringGuru\State\Conceptual\ConcreteStateA.
ConcreteStateA handles request1. ConcreteStateA wants to change the state of the
context. Context: Transition to RefactoringGuru\State\Conceptual\ConcreteStateB.
ConcreteStateB handles request2. ConcreteStateB wants to change the state of the
context. Context: Transition to RefactoringGuru\State\Conceptual\ConcreteStateA.
```

```
 &>my_files_+_John_J_Pruett_I_I_are_[Trust_Property]_of_[[Oracle_Enteprises]=[O
raclecloud.com]]_+_they_By_Authorisation=Power_of_Attorney_over_my_[Corporate_Tr
ust_with_15_%_ownership_=_entitlement_to_afore_mentioned_15_%_of_Total_allowed_C
apitol_gains_for_duration_of_this_[Contract=Smart_Contract]_._et_._al_with_[U_._
S_._Bitcoin_Bank_and_Exch_._et_._al_.]_+_[United_States_Bitcoin_Bank_and_Exchang
e_Limited_Liability_Company].et.al_[zachwylde_64]_+_[zachwylde_69]_+_[zachwylde_
70_,zachwylde_00_et_,_al_]_@_gmail_._com_+_@_icloud_._com=_my_[namespace]_Proper
ty_+_Data_Exclude:[Property]=[PhotoGraph_non_QRpng_non_value]_=_family_photos_th
ose_are_[Property]_=_[[[Namespace_[Brandy_Lynn_Houston]]_+_[AKA_,_Married_or_any
_other_Assumed_Name]]]]_only_[Property_in_any_account]_via_bankruptcy_9-28-
2017_mutexlock_(_2019/2017_05/10_08/28_,_15_:_32_:_599_Z_)_Copyright_Data_(_Excl
usion_effective_Date_=_[Bankruptcy_Date]_=_Data_Security_._Furthermore_[OracleEn
terprises]_with_only_mutually_benificial_actions_allowed_]_Auto-
Processes_all_my_to_current_date_as_paid_service_supplied_[my_namespace]_Payment
s_Expensis_Reciepts_,_Taxes_,_Income_Data_Processing_and_Dispersal_for_me_as_[[N
ameSpace=JohnJPruettII]]_._Appointed_:_[[[[[[[[Corporate_Bank_Board_Of_Directors
_Inclusion_Myself]_as_[Trust]_Include_:_[My_Accounts]=[Gains]To[Corporate_Trust_
Gains_Accounts]_as_[United_States_Bitcoin_Bank_and_Exchange_Limited_Liability_Co
rporation]_=_[NameSpace]_+_[NameSake]_=_[Inhereted_:_(_Seeds_,_Vessels_,_Account
s_and_Corporate_Political_Relations]]]]]]]]_entities—
bodies_within_my_authority_and_my_right_for_my_namesake.et.al)_furthermore_this_
does_not_allow_[OracleEnterpises]_all_[income=Gains]_rights_but_,_does_allow_sli
ghtly_above_avg_payment_for_[[[Services_Rendered]+[Afore_Mentioned_Property_as_[
(_%_of_)_=_Gains]]]_._
superuser_+_dev/not_null_is_[Infinisioso]_is_True_._Signed_John_Jeffrey_Pruett_I
I_Date[October_01_,_2017]
```

```
 BITBANK&1. GQ.1981.INC 1971.HQ
```

```
  [REGISTEREDUSERJOHNJPRUETTII]
```

```
      ns77.domaincontrol
```

```
   THIS MEANS NO PEAKING
```

MutexLock=(2017-10-01,17:11:399Z)USCST

MutexLock(201905091726599Z)USCST

Mutexlocktime:(2017-10-01:15:55:599Z)USCST

(v1.2)]<TARGET>|BLOCK ANY UNAUTHORIZED_ACCESS+THEFT+IDENTITY MISUSE/ERROR,INCLUDE THIS PROGRAM_HAS_AUTHORIZED ACCESS TO:WALLETSERVICE+WALLETSERVICE PERMISSIONS AND SETTINGS_FOR_THIS_DEVICE

Function:/ACCESS+FUNCTION ENABLE_RESOURCES +COMPLETE REQUESTS VIA WALLETSERVICE_

SUBJECT:THESE"PERMISSIONS"+"SETTINGS" AUTOAPPLIED TO EVERY DEVICE WITH AUTHORIZED ACCESS=DEPENDENCIES MANDITORY:1)HAVE THE PERMISSIONS SOMEWHERE IN FILE ON DEVICE

2)ENTER CORRECT_WALLETSERVICE_ NUMBER AS EMAIL

3)REÇIEVE ACCESs GRANTED VIA "usbitcoinbankandexchange@gmail.com"

once you have received email all permissions will be auto installed/autoaccessible via device user settings+'all' permissions and settings will be active and autoenforced to allowed 'device 'regardless of native root via WiFi is WiFi internet/Bluetooth connectivity

failure of completion or compliance to these mandatory actions and requests properly via correct paths will result in autodestruction of attempting device

compliance and completion time set at two mins to complete timer kept by national time weather clockzone=USCST

SEND VIA EMAIL; REQUEST=[[AUTO-REPAIR,AUTO-CORRECT,AUTO-FIX,AUTOENABLE,AUTOCOMPLETE,AUTOCORRECT(ANY/ALL_ STORAGE,NAMESPACE,DB,DATABASE)ERRORS+ISSUES+SERVICE_BLOCKS+MISSING DATA+INCORRECT DATA=ENABLE_AUTO-POST (AMOUNTS+VALUES+BALANCES)TO JOHNJPRUETTII(NAMESPACE_ACCOUNTS_ MAKING_BALANCES_"ACCESSABLE/SPENDABLE"TO/BY:[JOHN J PRUETT II']]|<TARGET>

FUNCTION:MANDATORY ACTION 'REQUIRED'PROGRAM COMPLIANCE=[COMPLETE ALL REQUEST+TARGETS]IS TRUE+ABSOLUTE

REQUIRED_MUST_COMPLY_IS" MANDATORY_ACTION_REQUIREMENT ALL WALLETSERVICE_SETTINGS=AVAILABLE+ACTIVE IN MY DEVICE "UI"AND SETTINGS

PERMISSIONS='GRANTED'+ENABLED

ALL MY REQUESTS_AUTHORIZATIONS='AUTHORIZED'

SUPERUSER_ADMINSTRATOR_PERMISSIONS+SETTINGS=ACTIVE/AVAILABLE/APPLIED TO THIS DEVICE

open settings apply "user"="DB_ADMINISTRATOR"

mutexlock(201905052016599Z)

CHANGES+RUN PROGRAM FORMAT "JSON" RUN THIS "COMPUTER"=(ONLY COMPUTER ALLOWED IN PROGRAM+'EXCEPTION' ALLOWED APIS AND NECESSARY INTERACTIONS WITH ALL SYSTEMS MONITORING BLOCKCHAIN,'ALLOW='NECESSARy FOR OUTPUT_VALUE')='MANDATORY_FUNCTION OF_THIS SYSTEM AND FILE;' Typesprivate/public block chain secure connection to mainchainviainclude:[appID "wallet service"

FUNCTION:/GET PRE TRANSACTED COINS POST TO:CORRECT PATH IF MISSING RECREATE IF LOCATED AFTER RECREATION REPLACE RECREATES WITH ORIGINAL AUTO UPDATE DATABASE

THEN REMOVE RECREATION FROM DATABASE POST TIMESTAMPS FOR EVERY ACTION TAKEN AND DO NOT REMOVE TIMESTAMP FROM DATABASE

INCLUDE ORACLE DATABASE INCLUDE XLDB DATABASE INCLUDE BITCOIN TRANSACTION DATABASE UPDATE+GET +POST ALL COINS MINED FROM RSK+COINBASE +BLOCKCHAIN+BTC.COM+SATURNNETWORK+XLDB

include my devices:"MY ELLIPSIS", MY "Moto_e4",MY "HTC_EXODUS","MY APPLE_IPAD",MY "MOTO_E5PLAY", ALL OTHERS ="UNAUTHORIZED"+REMOVE MY COINS +REMOVE MY DATA IF "DEVICE"HAS MY DATA PLUS COINS AFTER REMOVAL DISABLE UNAUTHORIZED DEVICE PERMANETELY +NO NOTICE OR AUTHORIZATION REQUIRED

Function:/include reporting all transactions + ANy other chain related' data' to US Bitcoin Bank and Exchange Private Blockchain/Public Blockchain at url="https://xooa.com/blockchain/ledger"

mute click(21905012239599'Z) url:'https://www.pwastore.com/w/walletservice-app]also googlepay[profile:johnpruett_0969-8556-3444_us/71291]

 [XLDB/INCLUDE[WALLETSERVICE-APP] #INCLUDE[SMARTCONTRACT-ETHEREUM I]VIA WWW.SATURNNETWORK.COM+[SMARTCONTRACT-BITCOINVIA BLOCKCHAIN,VIA:BCT.COM,VIABITSTAMP,VIA:BINANCE.COM,VIA;ALSO VIA:coinbase.com[Bitcoin:/mainchain.com]

#include:inclusion>[(Secure bank keys + secure server access API)

#include

MAINCHAIN+search/correct fork,check(reliability)=inclusion

mutexlock(201904291640599z)

Coinbase should #include:ebankintllc@gmail.com

To:usbitcoinbankexch@gmail.com

><value>

mutexlock(201904291646589Z)

Security certificate

GlobalSign Domain Validation CA..

ISSUED TO:

United States Bitcoin Bank and Exchange LLc.

Common name:USBitcoinBank&ExchangeLLc.

GlobalSign Domain Validation CA-SHA256

-G2

Organization:FinancialInstitutions

GlobalSign nv-sa

Organizational unit:SSH|SSL/w Encrypted storage

Root CA

Serial Number:04:00:00:00:00:01:44:4E:FO:3E:20

Issued by:

GlobalSign Domain Validation CA-SHA256-G5

Organization:USBitcoinBank&ExchangeLLc.

GlobalSign nv-sa

Organizational unit:Secured Encryption Root CA

Validity:

Issued on:2/20/14

Expires on:2/20/24

Fingerprints:

SHA-256 fingerprint:

BF:DF:4C:F3:F1:43:AD:0D:B9:12:D8:AB:3A:7C:12:F6:17:B9:EA:60:CE:

8B:1F:4E:44:F7:42:70:

FB:21:B1:9B

SHA-512 fingerprint:

a8:69:b2:c7:f8:23:38:

eb:a5:b3:65:0b:a4:3c:

72:e6:ca:94:f5:2a:31:

eb:81:57:39:2f:78:1c:

b6:ab:27:67:ad:a8:8a:

e8:e2:94:2d:1c:70:0f:

44:b7:9c:20:1c:35:c7:

00:60:c6:7f:c5:c3:3f:

a8:43:54:a4:9a:f9:92:

49:e9:c1:64:9d:83:d6:

06:bb:23:a2:37:90:27:

1e:ad:39:39:59:10:ee:

c9:7a:9d:c7:0f:80:82:

df:a5:cf:6a:6e:a2:7a:

c0:58:a9:7f:45:4a:71:

93:88:d5:01:f8:c4:d5:

2e:66:c5:f0:38:21:90:

61:28:be:8c:03:2d:8f:

c8:11

SHA-1 fingerprint:

73:6A:4D:C6:79:D6:82:

DA:32:15:63:64:7C:60:

F6:99:FO:DF:C2:68

=<value>

-----END CA SIGNATURE-----

□<

□Mutexlock (2019051004315992)

□BEGIN PUBLIC SIGNATURES:

□dash://xpub6CLF69gtyfqFat4sqd2w9XvLwe4uBiV9mHZCzckKvyJhd72agS1KCnZaEf2zQV4h8k8WCuFTphdkFfFQu6wfxw8gErLWTrv4KAHU5G6dv81/recieve_todash://

□XfNEgZra2gQ26kzgyzS4w8nTDBEc7hbFDx;bitcoincash://xpub6BjUvE9PHJwcMz1jJm7CcgwVtQzcFgrKftx8pVGQinDmxiwrRTSqRaeUSEuXgTS9V1zNjxrmAVdUPFmijY95ptYMHohX3NyhXjhXxNcYjmk/recieve_to:bitcoincash://

□qr3yyqgd6h8h3peqcgtdvq39m8y79ayedv28p6lvpt;Bitcoin:/zpub6qghgqyfrygkYLoB9sA39v2axcAewssc64ugN4tyW8MMWX61wZgj8PLQgULZKV2buwf8R4kpNJk8FNKJFaEBaVWutGM2sJ6sRwRz9Zr3k5v/recieve_to:Bitcoin://

□bc1q8fyn9s4eszr68aslllmta9wxa2ngfm7pmrgndl;litecoin://

zpub6rfN5vrwbQBw7Ugi3XgH3wDuZC3RawmD3XKRE6PTRwpH5sMSaS9Xmpk9BLC2tMXthP1BPC2W5V4kHgxXsLqpGPNA9djCgyb8CvuLCsyZfXs/recieve_to:litecoin://

ltc1qgccq2t8ymyct0muvn4k0v4t7q0kr0hhva775ch;

mutexlock (2019042903435992)US CSTJJP2

</data>] <inclusion Algorithms specification:

MD5 <http://www.ietf.org/rfc/rfc1321.txt>

 RIPEMD-160 <http://homes.esat.kuleuven.be/~bosselae/ripemd160.html>

 SHA1 <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>

 SHA256 <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>

 SHA512 <http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf>

 HMAC <http://www.ietf.org/rfc/rfc2104.txt>

SUBJECT:REFORMAT(JSON)RUN PROGRAM SERIALIZERS AND KEYS+ALL/ANY AVALIABLE RESOURCES TO<NEW-TARGET>WRITE_COINS+MINE_COINS<NEW-TARGET> FUNCTION:/ WRITE+BEGIN"MINING""UNITEDSTATESBITCOIN"[56000000.00000000USBCT]

whirlpool+sha256+sha384+include "reactive color enhanced hash"+"scenic QRPNG"+"WATERMARK"QRPNG"WITH FLOATING "GQ.1981.INC"OR "BITBANK&1"OR "ns77"=algorithm To:Create coins=coin:"United States Bitcoin"

#Include:

Arrange Block "inclusion"+ fork @hieght:575531 to:Include ["UnitedStatesBitcoin" abbreviated "USBTC"]MERGED FORK INCLUSIONvia"RSK"SMARTCONTRACT=BITCOIN_TO:" www.bitstamp.com,www.coinbase.com,www.binance.com,www.coindesk.com";

#INCLUDE:

HASH PRELEADINGCOUNT="000000000000000000"

hash=4cbc94468e50bd4b2c07d17d817d6f63ee9ff71bf8fe514ca6

Hexidecimal-of-hash"

52 99 98 99 57 52 52 54 56 101 53 48 98 100 52 98 50 99 48 55 100 49 55 100 56 49 55 100 54 102 54 51 101 101 57 102 102 55 49 98 102 56 102 101 53 49 52 99 97 54 "

Binary-of-hash"

□00000000000000000000000000110100 00000000000000000000000001100011
00000000000000000000000001100010 00000000000000000000000001100011
00000000000000000000000000111001 00000000000000000000000000110100
00000000000000000000000000110100 00000000000000000000000000110110
00000000000000000000000000111000 00000000000000000000000001100101
00000000000000000000000000110101 00000000000000000000000000110000
00000000000000000000000001100010 00000000000000000000000001100100
00000000000000000000000000110100 00000000000000000000000001100010
00000000000000000000000000110010 00000000000000000000000001100011
00000000000000000000000000110000 00000000000000000000000000110111
00000000000000000000000001100100 00000000000000000000000000110001
00000000000000000000000000110111 00000000000000000000000001100100
00000000000000000000000000111000 00000000000000000000000000110001
00000000000000000000000000110111 00000000000000000000000001100100
00000000000000000000000000110110 00000000000000000000000001100110
00000000000000000000000000110110 00000000000000000000000000110011
00000000000000000000000001100101 00000000000000000000000001100101
00000000000000000000000000111001 00000000000000000000000001100110
00000000000000000000000001100110 00000000000000000000000000110111
00000000000000000000000000110001 00000000000000000000000001100010
00000000000000000000000001100110 00000000000000000000000000111000
00000000000000000000000001100110 00000000000000000000000001100101
00000000000000000000000000110101 00000000000000000000000000110001
00000000000000000000000000110100 00000000000000000000000001100011
00000000000000000000000001100001 00000000000000000000000000110110 "

□TRANSACTION KEY:"

□NGNiYzk0NDY4ZTUwYmQ0YjJjMDdkMTdkODE3ZDZmNjNlZTlmZjcxYmY4ZmU1MTRjYTY&amount=[560
00000.00000000USBTC]"

□FUNCTION:/"AutoTimestamp" (completion of Coin Creation+every
transaction)="MandatoryFunction/Action"

□Type:"STABLECOIN"

Coin Value Source:"[73%_of_U.S._ Bitcoin_Bank_and_Exchange's_Total Net_worth&amount_in_bitcoin]"

Difficulty to write coins as an association for comparison:"56,000,000,000,000"

amount of Total Coin production:"56,000,000"

length of time taken to write coins:"560 weeks"

Coin Algorithm:"whirlpool+sha256+sha512"

Coin is not"premined"

Mining source and Contract:"U.s.BitcoinBankandExchangeLLc"+("RSK")MERGED VALUE BITCOIN+INVESTED VALUE BACKED IN "BITCOIN"

NATIVE COIN CONVERSION_CURRENCY:"UNITED STATES DOLLAR"

ISSUER OF COINS:COINS TO BE AUTO-WRITTEN AND STORED BY"US BITCOIN BANK AND EXCHANGE" THEN AUTO-DELIVERY UPON COMPLETION TO: "THE UNITED STATES DEPARTMENT OF TREASURY" FOR USE

 UnitedStatesSecureBitcoin(rsk) To be shield signature Bitcoin secured value@&&Bitcoin issue amount 56,000,000.00@5.6million Mined per yearUnitedStatesBitcoin Secured for American public and usbitcoinbank and exchange for no fee but at a rate7% for us Bitcoin Bank and exchange,20%for US Government+tax for the 73%to be donated via us government who will receive 93% of total PRODUCED UNITEDSTATESBITCOINS COINS TO BE RELEASED IMMEDIATELY, VIA BACKED VALUE INVESTED IN CREATION BY " US BITCOIN BANK AND EXCHANGE LLc"+AUTO-DELIVERRD TO:"UNITED STATES DEPARTMENT OF TREASURY"_VIA_SECURED_SERVER PATH =(IF_ALLOWED+IF_LEGAL)=API:"WWW.UNITEDSTATESDEPARTMENTOFTHETREASURY.GOV/RECIEVE/UNITEDSTATESBITCOIN" UPON DELUVERY "USBITCOINBANKANDEXCHANGE"WILL RECIEVE "7%" OF TOTAL PRODUCTION +BEGIN AUTO MINING COINS FOR ITS RETURN OF IT'S [NAMESPACE]INVESTED 'VALUE' THIS ALLOWS FOR IMMEDIATE RELASE AND INVESTED VALUE CARRIES VALUE NECESSARY FOR COIN IMMEDIATE MARKET VALUE APPROXIMATE VALUE IS UNKNOWN TO BE CALCULATED UPON COMPLETION AND SHOWN IN LOGS AND IN FILE AT END OF THIS PROGRAM RUN TO CREATE/WRITE COINS

SIGNED:[REGESTEREDUSERJOHNJPRUETTII]

mutexlock(2019051103005997)USCST

from usbitcoinbankand exchange LLc VIA [ORACLEENTERPRISES] &["RSK"SMART_BITCOIN_CONTRACT] EQUALS=[42MILLION]PRODUCED DURING TEN YEAR TERM PRODUCTION OF [UNITEDSTATESBITCOIN] MINED SOLEY BY [UNITED STATES BITCOIN BANK AND EXCHANGE LLc],+((93%of [namespace value](to back USBitcoin value)),for the rate of 7% of total value and for the betterment of futures for America and Americans mutexlock(20190510060499Z)USCST

dependencies include bridging economic gaps to help eliminate poverty and hunger +housing and medical care first and foremost priority. second set to help pay college tuitions third help elderly any purpose deemed necessary for "pursuit of happiness"hi

curl application interface usbitcoimbankexchllc';{

web API include SSH include: (encryptions)RSA,DES,ECDSA,DSA,

include:(hashing algorithms)"sha256","Sha1","Sha2","sha3","sha512","sha256+keccak256","keccak","lyre","Sha2+nistp521" "base64 runtime protocol"in accordance to FCC regulation bank standards and practices web implentation of secure hashes transmissions within"United States of America"

include:

Title-"U.S.BitcoinBankandExchangeLLc"

Namespace-"John J Pruett II

device-"Moto e4","Moto e5play","Dell laptop","htc Exodus 1","ellypsis tablet","iPad mini"

date-11/28/2017-current*(note previous apis are available upon request)

server1-https://la.h.75.65.204.177@comcast.net

backup server1-https://www.usbitcoinbankandexch@ns77.domaincontrol.com {

 }

server 2:program development secure hash transmission notification+notify;

"https://www.usbitcoinbankandexch.directory:"\hash.algorithms.api\","\

www.walletservice.java\"/\;/www.usbitcoinbankandexch.com/"\api.blockchain.info\","\www.trustwallet.com\","\www.token\assetplatform.net\","\www.federalbureauofinvestigation.gov\","\www.finra.org\","\www.sec.org\","\www.coincloud.com\","\www.ebankintllc@gmail.com\","\www.zendipper@yahoo.com\","\aws.com\","\www.godaddy.com\","\www.myetherwallet.com\","/www.la.75.65.204.177@comcast.net/"\www.oraclecloud.com\","\www.oracle.com\";

"\www.zachwylde00@gmail.com/JohnJPruettII/password=1Zr1Corvette&&$$\","\www.usbitcoinbankexch@gm:ail.com/JohnJPruettII/password=1Zr1Corvette&&$$\","\www.pruettiijohn@gmail.com\";/www.usbitcoinbankandexch.server.ns.77.domaincontrol.com;backup_server/www.ebank.us_ashford@oracle.com;,

curl HTTPS://www.usbitcoinbankandexch.server.ns1.digitalocean.com;"SSH_server"-HTTPS://www.usbitcoinbankandexch.server@secureserver.net:20;backup_SSH_server,API.v2/hash_mining_secure_hashtransmission+transactions.java|

□https://www.usbitcoinbankandexch.directory;/:\:/web_hash_algorithms_API.java/:/www.walletservice.java/:/:\api.blockchain.info\:\www.trustwallet.com\:\www.tokenassetplatform.net\:/www.federalbureauofinvestigation.gov/,/www.finra.org/:/www.sec.org/:/www.coincloudcom/:/www.ebankintllc@gmail.com/:\www.godaddy.com\:\www.digitalocean.com\:\www.zendipper@hotmail.com\:/aws.com/:\www.myetherwallet.com\:/www.oracle.com/:/www.oraclecloud.com/:/www.usbitcoinbankexch@gmail.com/:/www.pruettiijohn@gmail.com/user=JohnJPruettII/password=1Zr1Corvette&&$$/;\backupSSHserver2\:/www.usbitcoinbankandexch.server/@secureserver.net/;/SSH_server2/.ns.1.digitalocean.com

□/;. {

□ {

□}

□include:program development hash mining +securehash transmissions+transact API.v2hashtransmit/transact.java|

www.usbitcoinbankand exchange.directory,sub-
domain_www.usbitcoinbankandexch.com:

"www.walletservice.java",

"www.api.blockchain.info","www.trustwallet.com","www.tokenassetplatform.net","w
ww.federalbureauofinvestigation.gov","www.internalrevenueservice.gov","www.finra
.org","www.sec.org","www.louisianaofficeoffinancialinstitutions","www.coincloud.
com","www.ebankintllc@gmail.com","GQ.1981.INC","www.godaddy.com","www.zendipper@
yahoo.com","www.microsoft.asp.razor.ai.NET.COM""www.amazonaws..com","www.myether
wallet.com","www.saturnnetwork.com","www.comcast.net","www.entergy.com","www.ora
cle.com","www.oraclecloud.com","www.paypal.com","www.cashapp.com","bws.bitpay.co
m/API/0/44","www.BTC.com","www.BTC.top","www.coinbase.com","www

.binance.com","www.htc.com","www.tomas@rijndael-project.com","www

pruettiijohn@gmail.

com","www.usbitcoinbankexch@gmail.com"root
IP=127.0.0.1/rootwifIP=10.0.0.22:8222/backup_dnssupport_via_dynamicdnsupdater/ro
otserver2-"www.usbitcoinbankandexch.server.ns.3.digitalocean.com"

□;/server2_backup-"www.usbitcoinbankandexch.server@secureserver.net:20";{

□

□ {

□

□ {

□{

mutexlock(2019042605135992)

MUTEXLOCK(2019041705045992)

time-name="Package\WalletBalance\dimDate">
<Properties>XXGVAQBCKLUL2<Property>Blockchain<Name>DataSourceViewID</Name>XXGVAQ
BCKLUL2</Property>' Bitcoin' <Property> <Name>TableInfoObjectType</Name>BTC
Blocks with ',pow'<Value type="q2:string">Table</Value>hieght
</Property>BlockGenesis </Properties>decendents+
ancestry(NameSpace:zaçhwylde00@gmail.com,usbitcoinbankexch@gmail.com,johnjpruett
ii@usbitcoinbankexch.org,usbitcoinbankandexch@icloud.com</PipelineComponentMetad
ata>keys&amounts(value) </Objects>]]>serializedhash,blocked
blocks,3Dvisual,coined</DTS:DesignTimeProperties>time of creation
'timestamp'</DTS:Executable>current'keys/amount=include=QRpng/objects+Post

mutexlock(2019040602035992)

SUBJECT:' WALLET
'URL'='https://www.usbitcoinbankandexch.server@ns77.domaincontrol.com'

□'SSH'PROTOCOL='p3plssltools01.cloud.phx3.gdg 1.3.1 290831b 4/9/2019 10:27 pm -07:00'

□'CONNECTION'='IPV6:

□(2601:3c6:4280:1264:7ca3:de60:5679c1a2]

□IPV6: [2601:3c6:4280:f264 debf e9ff.fedb:

□8b90]

IPv6: [fe80::deef.e9ff feb:8b90%wlano)'=U.S.BitcoinBankandExchangeLLc'='WEB TRUST AGENT'GLOBALSIGN CA AUTHORITY ROOTCA SERVICE'='A.I.'RESOURCE=WRITE,TRANSACT,COMPLETE'ALL'TASKS INCLUDE: 'FILES' LISTED+UPDATE AMOUNTS(report'ONCE ONLY')useRESOURCES>Oauth 2.0/REST/API/AUTHORISATION OVERRIDE TOKEN BEARER:eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJBcGlLZXkiOiJOTU5XVldRLUtIOTRNNEQtUDZFUTY1Vi1UM040RUs3IiwiQXBpU2VjcmV0IjoiR2VUN3djdWpkOUthakplIiwiUG

[reportassets\bip39assets\bip39-wordlist.txt assets\checkpoints-testnet.txt assets\checkpoints.txt assets\ssl-keystore Raw File Namesres\raw\coins_received.wav res\raw\send_coins_broadcast_1.wav res\raw\send_coins_broadcast_2.wav res\raw\send_coins_broadcast_3.wav res\raw\send_coins_broadcast_4.wav)+REPORT ALL INSTANCES WITH AMOUNT +POW TO:&>my_script file TO:&>my_script hidden=none,show=every,update=every permissions=True,echo,walletservice(permissions+settings)=true property+amounts+coins+all(addresses+nodes)posted to xpubs located in my_script=thisfile, file=private is true+ include:lockout/block(access via'remote'entry/hidden"setting",hidden"script",hidden"coding/code"+&>my_script Dev/not_null   mutexlock(201905051843599Z)

 (2019040611106599Z)(2019040617045997)

package com.ui.androidwalletservice.java;

Android.permissions.ACCESS_WINDOWS_FILES

Android.permissions.ANY/ALL_DATA/FILES/SETTINGS/PASSWORD_UNVIEWABLE/HIDDEN/NO_ACCESS_ANY_OTHER _USER_ANY_OTHER_DEVICE=NOT_MY_DEVICE_NOT_MY_NAMESPACE

Android.permissions.AUTOACCESS_ANY/ALL_DATA_VIA_MY_NAMESPACE_LOGGEDIN/LOGIN+MY_WALLETSERVICE+MY_SOCIAL_SECURITY_NUMBER

Android.permissions.AUTOSYNC_ALL_DEVICES/ALL_APPLICATIONS/ALL_SETTINGS/ALL_WALLETS_VIA_WIFI/INTERNET/BLUETOOTH_EVERY_FIVE_MINUTES

Android.permissions.AUTODELETE_MY_DATA_ANY_OTHER_USER'S_DEVICE_UPON_DETECTION

Android.permissions.AUTOSCAN_ANY/ALL_OTHER_USER'S_DEVICES_FOR_MY_DATA_VIA_WIFI/INTERNET/BLUETOOTH/APPLICATION/APK_EXCEPTION=ORACLE_ORACLECLOUD_AWS_XOOA_XLDB_MY_EMAIL_MY_NAMESPACE+LOGGED IN

Android.permissions.ACCESS/MODIFY/CONFIGURE/ADD/DELETE_OTHER_USER'S_SETTINGS/DATA/PERMISSIONS_VIA_HAS_MY_DATA_EXCEPTIONS=MY_API+LISTED_AS_ALLOWED_MY_DATA

Android.permissions.ACCESS/STORE_ANY/ALL_CODE/CODING/ENCODING+ANY/ALL_DECODING/KEY/ALGORITHM_AS_RESOURCE

Android.permissions.ACCESS_MY_SCRIPTS_ANY/ALL/EVERY_DEVICE

Android.permissions.AUTOCORRECT_ANY/ALL_SYNTAX/ARGUMENT/COPYWRITE/TRANSACTION/ILLEGAL_ACTION/ERROR/MISSING_DATA/SYNTAX/ENCODING_MY_SCRIPTS+UPDATE_ALL_AUTOSYNC_ALL_AUTOCOMPLETE_ANY/ALL/EVERY

Android.permissions.ACCESS/MODIFY/CONFIGURE/ADD/DELETE_RAM_MEMORY/STORAGE/SDCARD

Android.permissions.ACCESS/RECONFIGURE_DEVICE_CORRECT/REPAIR_DEVICE_RAM/STORAGE/SDCARD=FIX_LOW_STORAGE/MEMORY_ERRORS/ISSUES_MAXIMIZE_DEVICE_EFFICIENCY_VIA_AUTOSERIALIZE/AUTODESERIALIZE_DATA_WITHOUT_LOSS/ERROR/DYSFUNCTION_TO_DEVICE_OR_DATA

Android.permissions.ACCESS/AUTODELETE/AUTOBLOCK_MALEWARE/SPYWARE/AD/AD_SETTING/HIDDEN_APPLICATION_AD_SETTING/HIDDEN_EMAIL/EMAIL_ACCESS_SETTING_WITHOUT_LOSS_OF_APPLICATION/EMAIL

Android.permissions.READ/MODIFY/CONFIGURE/DELETE/ADD_WINDOWS_FILES

Android.permissions.ACCESS_ANY/ALL_SQL_DATABASE

Android.permissions.ACCESS/MODIFY/ADD/CONFIGURE/DELETE_ANY/ALL_SQL_DATABASE_SETTINGS

Android.permissions.ACCESS/MODIFY/ADD/CONFIGURE/DELETE_TRANSACTIONS_SQL_DATABASE

Android.permissions.ACCESS_ANY/ALL_SUPERUSER_AUTHORITIES_PERMISSIONS

Android.permissions.ACCESS_ANY/ALL_DATA_AS_SUPERUSER_ADMINISTRATOR

Android.permissions.SUPERUSER_ACCESS_BYPASS_ANY/ALL_PASSWORD_PROOTECTED_PATH_IN_RELATION_TO_MY_DATA

Android.permissions.ACCESS_MY_DATA_TRANSACTIONS_VIEW/MODIFY/CONFIGURE/ADD/DELETE_TRANSACTION_CREATED_BY_ANOTHER_USER_WITHOUT_NOTIFICATION_OR_CONSENT_REQUIRED/GIVEN_TO_OTHER_USER+AUTODENY_DATA_ACCESS_TO_OTHER_USER

Android.permissions.ACCESS_SQL/LISTENER/JAVA_SETTINGS_AS_SUPERUSER_ADMINISTRATOR

Android.permissions.VIEW/MODIFY/CONFIGURE/ADD/DELETE_SQL/LISTENER/JAVA_SETTINGS/PERMISSIONS/ACCESS

Android.permissions.ACCESS/VIEW/MODIFY_ALL_SUPERUSER_SETTINGS/PERMISSIONS/DATA_AS_SUPERUSER_ADMINISTRATOR

Android.permissions.AUTODECODE/ACCESS/VIEW_ANY/ALL_DATA/FILE_ANY/ALL_CODE/ENCODING_ANY/ALL_OPERATING_SYSTEM_AS_SUPERUSER_ADMINISTRATOR

Android.permissions.ACCESS/USE/ENABLE_WALLETSERVICE_PERMISSIONS_ON_ANY/ALL_MY_DEVICES+AUTOENABLE_PERMISSION_UPON_DETECTION_OF_MY_DEVICES

Android.permissions.AUTODETECT_MY_DEVICES_VIA_WIFI/INTERNET_CONNECTION_THRU_WALLETSERVICE_APP/APK

Android.permissions.AUTOUPDATE_THESE_PERMISSIONS_IN_THIS_EMAIL_TO_WALLETSERVICE_APP+APK+WALLETSERVICE_ON_MY_DEVICES

Import.os.ANDROID_PERMISSIONS.WalletServicePermissions.java

Import.java.WALLET_SERVICE_PERMISSIONS.OS_10.5

import.java.util.ArrayList;

import.java.util.List;

import.java.util.AndroidPermissions;

import.android.content.ContentValues;

import.android.content.Context;

import.android.WalletServicePermissions;

import.android.database.Cursor;

import.android.database.sqlite.SQLiteDatabase;

import.android.DBManagerScriptCliplet=[[[{

```
public class DBManager {

    private SQLiteDatabase db;

    public DBManager(Context context) {

        DBHelper helper = new DBHelper(context);

        db = helper.getWritableDatabase();

    }

    public void add(List<AppRecord> recordList) {

        db.beginTransaction();

        try {

            for (AppRecord record : recordList) {

                db.execSQL("INSERT INTO apps VALUES(null, ?, ?, ?)",
```

```
                        new Object[]{record.packageName, record.label,
record.apkPath});

            }

            db.setTransactionSuccessful();

        } finally {

            db.endTransaction();

        }

    }

    // Add a record

    public void add(AppRecord record) {

        db.execSQL("INSERT INTO apps VALUES(null, ?, ?, ?)",

                new Object[]{record.packageName, record.label,
record.apkPath});
```

```
    }

    // Update record according to package name

    public int updateRecord(AppRecord record) {

        ContentValues cv = new ContentValues();

        cv.put("label", record.label);

        cv.put("apk_path", record.apkPath);

        return db.update("apps", cv, "package_name = ?",

                new String[]{record.packageName});

    }

    // Remove record according to package name

    public int deleteOldRecord(String packageName) {
```

```
        return db.delete("apps", "package_name = ?",
                new String[]{packageName});
    }

    // Query all the records
    public List<AppRecord> query() {
        ArrayList<AppRecord> recordList = new ArrayList<AppRecord>();
        Cursor c = db.rawQuery("SELECT * FROM apps", null);
        while (c.moveToNext()) {
            AppRecord record = new AppRecord();
            record._id = c.getInt(c.getColumnIndex("_id"));
            record.packageName = c.getString(c.getColumnIndex("package_name"));
```

```
            record.label = c.getString(c.getColumnIndex("label"));

            record.apkPath = c.getString(c.getColumnIndex("apk_path"));

            recordList.add(record);

        }

        c.close();

        return recordList;

    }

    // Close database

    public void closeDB() {

        db.close();

    }
```

```
}]]];


package.com.ui.WalletService.java;


package.ui.WalletServiceWallet.java;


import.android.WalletServiceWallet.java;


import.package.com.ui.AutoSyncAllApllicationsWalletsDataWithWalletServiceWallet
+WalletServiceData.java;


import.android.app.WalletServiceActivity;


import .android.app.WalletServiceWalletActivity;


import.android.content.pm.ApplicationData;


import android.os.Bundle;


import android.view.View;


import android.widget.Toast;
```

import java.util.List;

public class LocateCompletePostTra

mutexlock (201905120848599z)USCST

Android.permissions.ACCESS_ANDROID_SETTINGS_DATABASE

Android.permissions.MODIFY_CONFIGURE_DELETE_ANDROIDSETTINGS_DATABASE

Function:/repair Android settings via this file_permissions=correct settings make paths Key output(amounts+values+ balances+accounts)post To:namespace-[JOHNJEFFREYPRUETTII]ID#:005837703LADL

SOCIAL SECURITY:433-75-9830

ADDRESS:410 Gulpha Dr.

CITY:WESTMONROE

zip:71291

XPUB/PUBLICKEY:"LISTED IN THIS FILE;ALSO CONFIGURE DATABASE SETTINGS TO LOCKOUT ENTRY TO MY_ANDROID FILES+SETTINGS VIA HIDDEN/UNHIDDEN(CODING,SCRIPT,SETTING)

REPAIR ALL OUTSIDE ANDROID SOURCE OF NAMESPACE ATTACK +HACK+THEFT ON OR TO FILES +ACCOUNTSutexlock(201905051859599z)

import java.util.ArrayList;

```
import java.util.List;

import android.content.ContentValues;

import android.content.Context;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

public class DBManager {
```

```
    private SQLiteDatabase db;

    public DBManager(Context context) {

        DBHelper helper = new DBHelper(context);

        db = helper.getWritableDatabase();

    }

    public void add(List<AppRecord> recordList) {
```

```
        db.beginTransaction();

        try {

            for (AppRecord record : recordList) {

                db.execSQL("INSERT INTO apps VALUES(null, ?, ?, ?)",

                        new Object[]{record.packageName, record.label,
record.apkPath});

            }

            db.setTransactionSuccessful();
```

```
        } finally {

            db.endTransaction();

        }

    }

    // Add a record

    public void add(AppRecord record) {
```

```
        db.execSQL("INSERT INTO apps VALUES(null, ?, ?, ?)",

                new Object[]{record.packageName, record.label,
record.apkPath});

    }

    // Update record according to package name

    public int updateRecord(AppRecord record) {

        ContentValues cv = new ContentValues();
```

```
        cv.put("label", record.label);

        cv.put("apk_path", record.apkPath);

        return db.update("apps", cv, "package_name = ?",
                new String[]{record.packageName});
    }

    // Remove record according to package name
```

```
    public int deleteOldRecord(String packageName) {

        return db.delete("apps", "package_name = ?",

                new String[]{packageName});

    }

    // Query all the records

    public List<AppRecord> query() {
```

```
        ArrayList<AppRecord> recordList = new ArrayList<AppRecord>();

        Cursor c = db.rawQuery("SELECT * FROM apps", not_null);

        while (c.moveToNext()) {

            AppRecord record = new AppRecord();

            record._id = c.getInt(c.getColumnIndex("_id"));

            record.packageName = c.getString(c.getColumnIndex("package_name"));

            record.label = c.getString(c.getColumnIndex("label"));
```

```
            record.apkPath = c.getString(c.getColumnIndex("apk_path"));

            recordList.add(record);

        }

        c.close();

        return recordList;

    }
```

```
    // Close database

    public void closeDB() {

        db.close();

    }
}package com.ui.androidwalletservice.java;

import android.app.Activity;
```

```
import android.content.pm.ApplicationInfo;

import android.os.Bundle;

import android.view.View;

import android.widget.Toast;

import java.util.List;

public class LocateCompletePostTransaction extends Activity implements
View.OnClickListener {
```

```
    private DBManager dbManager;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        this.dbManager = new DBManager(this);
```

```
        initView();

    }

    @Override

    protected void onDestroy() {

        dbManager.closeDB();

        super.onDestroy();
```

```
    }

    private void initView() {

        findViewById(R.id.btn_add).setOnClickListener(this);

        findViewById(R.id.btn_remove).setOnClickListener(this);

        findViewById(R.id.btn_revise).setOnClickListener(this);

        findViewById(R.id.btn_query).setOnClickListener(this);

    }
```

```
    @Override

    public void onClick(View view) {

        int id = view.getId();

        if (id == R.id.btn_add) {

            addRecord();

        } else if (id == R.id.btn_remove) {
```

```
            removeRecord();

        } else if (id == R.id.btn_revise) {

            reviseRecord();

        } else if (id == R.id.btn_query) {

            queryRecords();

        }
```

```
    }

    private void queryRecords() {

        List<AppRecord> records = dbManager.query();

        StringBuilder sb = new StringBuilder();

        sb.append(String.valueOf(records.size()));

        sb.append(" Records");
```

```
        if (!records.isEmpty()) {

            sb.append(":\n");

        }

        for (AppRecord rec : records) {

            sb.append(rec.label);

            sb.append("\n");

        }
```

```
        Toast.makeText(this, sb.toString(), Toast.LENGTH_LONG).show();

    }

    private void reviseRecord() {

        AppRecord record = new AppRecord();

        ApplicationInfo appInfo = getApplicationInfo();

        record.packageName = appInfo.packageName;
```

```
        record.label = "Revised";

        record.apkPath = appInfo.sourceDir;

        int revised = dbManager.updateRecord(record);

        String message = String.format("Revised %d record(s)", revised);

        Toast.makeText(this, message, Toast.LENGTH_SHORT).show();

    }
```

```
    private void removeRecord() {

        int deleted = dbManager.deleteOldRecord(getPackageName());

        String message = String.format("Removed %d record(s)", deleted);

        Toast.makeText(this, message, Toast.LENGTH_SHORT).show();

    }

    private void addRecord() {
```

```
        try {

            AppRecord record = new AppRecord();

            ApplicationInfo appInfo = getApplicationInfo();

            record.packageName = appInfo.packageName;

            record.label =
getPackageManager().getApplicationLabel(appInfo).toString();

            record.apkPath = appInfo.sourceDir;
```

```
            dbManager.add(record);

            Toast.makeText(this, "Added a record", Toast.LENGTH_SHORT).show();

        } catch (Exception e) {

            Toast.makeText(this, "Error: " + e.getMessage(),
Toast.LENGTH_LONG).show();

        }

    }
```

Function:/API 'data' for AutoComplete+AutoCreate+AutoPost run self check to:AutoCorrect 'missing'Data 'incorrect'Data+AutoCorrect

https://api.blockchain.info/v2/receive?xpub=$xpub&callback=$callback_url&key=$key

As defined in BIP 44, wallet software

https://api.blockchain.info/v2/receive?xpub=$xpub&callback=$callback_url&key=$key&gap_limit=$gap_limit

xpub - Your xPub (where you would like the payment to be sent)

callback_url - The callback URL to be notified when a payment is received. Remember to URL Encode the callback url when calling the create method.

key - Your blockchain.info receive payments v2 api key. Request an API key.

(Derive an unused address using your xPub:

curl
"https://api.blockchain.info/v2/receive?xpub=xpub6CWiJoiwxPQni3DFbrQNHWq8kwrL2J1
HuBN7zm4xKPCZRmEshc7Dojz4zMah7E4o2GEEbD6HgfG7sQid186Fw9x9akMNKw2mu1PjqacTJB2&cal
lback=https%3A%2F%2Fmystore.com%3Finvoice_id%3D058921123&key=[yourkeyhere]"

)

Response: 200 OK, application/json

{"address":"19jJyiC6DnKyKvPg38eBE8R6yCSXLLEjqw","index":23,"callback":"https://mystore.com?invoice_id=058921123"}

PHP Example

```
$secret = 'ZzsMLGKe162CfA5EcG6j'; $my_xpub = '{YOUR XPUB ADDRESS}'; $my_api_key = '{YOUR API KEY}'; $my_callback_url = 'https://mystore.com?invoice_id=058921123&secret='.$secret; $root_url = 'https://api.blockchain.info/v2/receive'; $parameters = 'xpub=' .$my_xpub. '&callback=' .urlencode($my_callback_url). '&key=' .$my_api_key; $response = file_get_contents($root_url . '?' . $parameters); $object = json_decode($response); echo 'Send Payment To : ' . $object->address;
```

///(Balance Updates [POST]

This method monitors an address of your choice for received and / or spent payments. You will be sent an HTTP notification immediately when a transaction is made, and subsequently when it reaches the number of confirmations specified in the request.

You are required to specify the request's notification behaviour. Setting the behaviour to 'DELETE' will delete the request after the first relelvant notification is sent to your callback address. Setting the behaviour to 'KEEP' will send additional notifications every time a transaction with the specified confirmations and operation type is sent to or from the address in the request.

Operation type is an optional parameter indicating whether the address will be monitored for received or spent transactions, or both. By default both operation types are monitored.

You may also optionally specify the number of confirmations a transaction reaches before being sent a notification. Note that you will receive a notification at 0 confirmations (i.e. immediately when the transaction is made), and again when it reaches the number of confirmations specified in the request (3 confirmations by default).

https://api.blockchain.info/v2/receive/balance_update

address - The address you would like to monitor

- callback - The callback URL to be notified when a payment is received.

- key - Your blockchain.info receive payments v2 api key. Request an API key.

- onNotification - The request notification behaviour ('KEEP' | 'DELETE).

- confs - Optional (Default 3). The number of confirmations the transaction needs to have before a notification is sent.

- op - Optional (Default 'ALL'). The operation type you would like to receive notifications for ('SPEND' | 'RECEIVE' | 'ALL').

Monitor an address for every received payment with 5 confirmations:

```
curl -H "Content-Type: text/plain" --data '{"key":"[your-key-
here]","addr":"183qrMGHzMstARRh2rVoRepAd919sGgMHb","callback":"https://mystore.c
om?invoice_id=123","onNotification":"KEEP", "op":"RECEIVE", "confs": 5}'
https://api.blockchain.info/v2/receive/balance_update
```

Response: 200 OK, application/json

```
{ "id" : 70, "addr" : "183qrMGHzMstARRh2rVoRepAd919sGgMHb", "op" : "RECEIVE",
"confs" : 5, "callback" : "https://mystore.com?invoice_id=123", "onNotification"
: "KEEP" }
```

The id in the response can be used to delete the request:

curl -X DELETE "https://api.blockchain.info/v2/receive/balance_update/70?key=[your-key-here]")////

Response: 200 OK, application/json

{ "deleted": true }

///(Block Notification [POST]

This method allows you to request callbacks when a new block of a specified height and confirmation number is added to the blockchain.

As with balance update requests, you are required to specify the request's notification behaviour to either 'KEEP' or 'DELETE'.

Height is an optional parameter indicating at which height you would like to receive a block notification - if unspecified, this will be the height of the next block to arrive.

Confs is another optional parameter indicating how many confirmations a block should have when a notification is sent.

https://api.blockchain.info/v2/receive/block_notification

callback - The callback URL to be notified when a block that matches your query is added.

key - Your blockchain.info receive payments v2 api key. Request an API key.

onNotification - The request notification behaviour ('KEEP' | 'DELETE).

confs - Optional (Default 1). The number of confirmations the block should have before a notification is sent.

height - Optional (Default current chain height + 1). The height at which a notification should be sent.

Request a single notification when the Bitcoin Blockchain reaches 500,000 blocks:

```
curl -H "Content-Type: text/plain" --data '{"key":"[your-key-here]","height":500000,"callback":"https://mysite.com/block?request_id=1234","onNotification":"DELETE"}' https://api.blockchain.info/v2/receive/block_notification
```

Response: 200 OK, application/json

{ "id" : 64, "height" : 500000, "callback" : "https://mysite.com/block?request_id=1234", "confs" : 1, "onNotification" : "DELETE" }

The id in the response can be used to delete the request:

curl -X DELETE "https://api.blockchain.info/v2/receive/block_notifcation/64?key=[your-key-here]")////

Response: 200 OK, application/json

{ "deleted": true }

////(Implementing the Callback

Receive and Balance Update callbacks

Please note, the callback url is limited to 255 characters in length.

When a payment is received by a generated address, or by an address monitored by a balance update request, blockchain.info will notify the callback URL you specify. For balance update callbacks and additional notification will be sent once the transaction reaches the specified number of confirmations.

transaction_hash - The payment transaction hash.

address - The destination bitcoin address (part of your xPub account).

confirmations - The number of confirmations of this transaction.

value - The value of the payment received (in satoshi, so divide by 100,000,000 to get the value in BTC).

{custom parameter} - Any parameters included in the callback URL will be passed back to the callback URL in the notification. You can use this functionality to include parameters in your callback URL like invoice_id or customer_id to track which payments are associated with which of your customers.

Block Notification callbacks

A block notification is sent every time a new block is added to the blockchain, and matches the height and number of confirmations set in the notification request.

hash - The block hash.

confirmations - The number of confirmations of this block.

height - The block height.

timestamp - The unix timestamp indicating when the block was added.

size - The block size in bytes.

{custom parameter} - Any parameters included in the callback URL will be passed back to the callback URL in the notification.

PHP Example

An example callback as a result of the above PHP example.

```
$real_secret = 'ZzsMLGKe162CfA5EcG6j'; $invoice_id = $_GET['invoice_id']; //invoice_id is passed back to the callback URL $transaction_hash = $_GET['transaction_hash']; $value_in_satoshi = $_GET['value'];[50000000000] $value_in_btc =[500] $value_in_satoshi / 100000000; //Commented out to test, uncomment when live if ($_GET['test'] == true) { return; } try { //create or open the database $database = new SQLiteDatabase('db.sqlite', 0666, $error); } catch(Exception $e) { die($error); } //Add the invoice to the database $stmt = $db->prepare("replace INTO invoice_payments (invoice_id, transaction_hash, value) values(?, ?, ?)"); $stmt->bind_param("isd", $invoice_id, $transaction_hash, $value_in_btc); if($stmt->execute()) { echo "*ok*"; })////
```

Expected<> Callback Response

In order to acknowledge successful processing of the callback, your server should respond with the text "*ok*" (no quotes), in plain-text, no HTML. If the server responds with anything else, or nothing, the callback will be resent again for every new block (approximately every 10 minutes) up to 1000 times (1 week). Callback domains which appear dead or never return the "*ok*" response may be blocked from the service.

Check xPub address gap [GET]

Check the index gap between last address paid to and the last address generated using the using the checkgap endpoint. Use the xpub you want to check and your API key like so:

curl "https://api.blockchain.info/v2/receive/checkgap?xpub=[xpub6DR7b7SxrbW3RA9JoGsbprtNdwFUMksmv8wX1H6L1k8Mwn4bS5AFAp6acT2VfSBqQ5q8nvYZjqmG6gymxxYEHa8LjtQhreoNtC3UpRLjXgY]]&key=[1FiFa1Qc12kRwtTRxPpQQdR2Xpr7Bzxunk]"{"gap":2}

Callback Logs [GET]

See logs related to callback attempts using the callback_logs endpoints. Use the exact callback in question and your API key like so:

curl "https://api.blockchain.info/v2/receive/callback_log?callback=https%3A%2F%2Fmystore.com%3Finvoice_id%3D05892112%26secret%3DZzsMLGKe162CfA5EcG6j&key=[yourkeyhere]" [{ "callback": "https://mystore.com?invoice_id=058921123&secret=ZzsMLGKe162CfA5EcG6j&key=[yourkeyhere]", "called_at": "2015-10-21T22:43:47Z", "raw_response": "*bad*", "response_code": 200 }, { "callback": "http://mystore.com?invoice_id=058921123&secret=ZzsMLGKe162CfA5EcG6j&key=[yourkeyhere]", "called_at": "2015-10-21T22:43:55Z", "raw_response": "*bad*", "response_code": 200 }]}

}

}

TppID "wallet service" url:'https:/

include: database file write plus keep file updated via (this file only)read+write +rewrite capable for reel ability across regions+

write file for wallet service reliable file = www.walletservice.java

USED PERMISSIONS IN FILE Android.permission.AUTOCORRECT_RUNTIME_ERRORS.SCRIPT.EDITING.java

android.permission.AUTOCORRECT_INCORRECT_SETTING

□android.permission.AUTOCORRECT_INCORRECT_SYNTAX.SCRIPT.EDITING.java

□android.permission.CAMERA

□android.permission.VIBRATE

□android.permission.RECORD_AUDIO

□android permission.ACCESS_WIFI_STATE

□com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE

com.google.android.c2dm.permission.WRITE_GET_RECEIVE_POST

android.hardware.camera autofocus

android.permission.READ_EXTERNAL_STORAGE

android.permission CAMERA

android permission.FLASHLIGHT

android.permission.CHANGE_WIFI_STATE

□android.permission.BLUETOOTH_ADMIN

□android.permission.CHANGE_NETWORK_STATE

□android.permission.MODIFY_AUDIO_SETTINGS

□android.permission.PACKAGE_USAGE STATS

□com.zachwylde00@gmail.ACCESS_LOGS_AS_SUPERUSER_INNER_BROADCAST

□com.google.android.c2dm.permission.RECEIVE_CERTIFICATE_USER_PERMISSIONS

com.android.launcher.permission.READ SETTINGs

com.htc.launcher.permission READ.SETTINGS

com.htc.launcher.permission.WRITE SETTINGS

com.oppo.launcher.permission.WRITE SETTINGS

com.oppo launcher.permission.READ SETTINGS

android.permission.SYSTEM.OVERLAY. WINDOW

□android.permission.READ_EXTERNAL_STORAGE

□android.permission.READ_CONFIGURE_MODIFY.APN.SETTINGS

□android.permission BROADCAST_STICKYCERTIFICATE.USED PERMISSIONS

□Android.permission.READ_CONFIGURE_MODIFY.VPN.SETTINGS

□OV

□android permission.ACCESSFINELOCATION

- android permission.ACCESS COARSE LOCATION
- android permission.BATTERY STATS
- android.permission BLUETOOTH
- Android.permission.CREATE_TASK
- android.permission.GET_TASKS
- com.android.launcher.permission.INSTALL SHORT

□CUT

□com.android.launcher.permission.UNINSTALLSHORTCUT

□com.android.launcher.permission.READ SETTINGS

□com.android.launcher.permission.WRITE_SETTINGS

□android.permission.SYSTEM_ALERT_WINDOW

□android.permission.ACCESS_WIFI_STATE

- android.permission.READ_LOGS
- android.permission.GET_PACKAGE SIZE
- android.permission.WAKE LOCK
- android.permission.WRITE SETTINGS
- android.permission.KILL_BACKGROUND_ PROCESS
- ES

- android.permission.INSTALL_PACKAGES
- android.permission.DISABLE_ KEYGUARD
- android.permission.REBOOT_IPO
- android.permission.INTERNET
- android.permission.VIBRATE
- android.permission.WRITE_EXTERNAL_STORAGE
- android.permission.RECEIVE_BOOT_COMPLETED

□android.permission.READ_PHONE_STATE

□Android.permission.ACCESS_CONFIGURE_MODIFY_HIDDENSETTINGS.SETTINGS

□android.permission.ACCESS.NETWORK.STATE

□Android.permission.MINE_VARIANT_CRYPTOCURRENCIES_ACCESS_RESOURCE_HASH_ALGORITHI
MS

□Android.permission.CONVERT.HASH_TRANSACT_HASH_POST_HASH

□Android.permission.AUTO_CORRECT_SCRIPTERRORS

Android.permission.BIND_AUTOTRANSACT_OVERRIDE_CURRENT_LISTENER_DATAIF_NECESSARY

android.permission. MOUNT_UNMOUNT_FILE SYSTEMS

MS BC

Android permission.DECODE_WINDOWSOS_FILES

Android.permission.AUTODECODE_CRYPTICKEYS_FOR_CORRECT_PATH_LOCATION_SERVICE

Android.permission.DECODE_RE-
ENCODE_KEYS_WITH_VALUE_PATH_LOCATION_SERVICE_CORRECTION

Android.permission.AUTOCORRECT_CODEDKEY_PATH

Android.permission.IMPORT_ALL_USERAPPLICATION_PUBLIC+PRIVATE_KEYS

Android.permission.ACCESS_USER_APPLICATIONS

Android.permission.AUTOCORRECT.LOW_STORAGE_ERRORS_ISSUES

Android.permissions.DELETE_HIDDEN_USELESS_UNNECESSARY_DATA+FILES.SETTINGS

Android.permission.RESTORE_DELETED_FILES.SETTINGS

Android.permission.ACCESS_CONFIGURE_MODIFY_USER_APPLICATIONS_SETTINGS

Android.storage/emulated/0.walletservice.KEY_STORAGE_FILE.data.javaappwidget_dark_bg_focused.9.png

appwidget_dark_bg_focused.9.png

appwidget_dark_bg_pressed.9.png

currency_symbol_btc.png

currency_symbol_btcg.png

currency_symbol_btcd.png

currency_symbol_usd.png

currency_symbol_bch.png

currency_symbol_eth.png

currency_symbol_bbtc.png

currency_symbol_xlm.png

currency_symbol_usbtc.png

dropdown_ic_arrow_small_focused_holo_light.png

dropdown_ic_arrow_small_normal_holo_light.png

dropdown_ic_arrow_small_pressed_holo_light.png

@gmail.com>

Correct run to me 'errors'+fill in missing data if exists+correct any/all "input"includeCorrections "remove hidden (requests/commands/text/syntax)

"Correct (incorrect/missing/blank+necessary)data+ replace with corrected",also "Complete (smart mining contract targets)"

John Pruett II <pruettiijohn@gmail.com> Sun, Apr 21, 2019 at 8:19 PM

To: usbitcoinbankexch@gmail.com

*

 The MIT License (MIT)

 Copyright (c) [2016] [BTC.COM]

 Permission is hereby granted, free of charge, to any person obtaining a copy

 of this software and associated documentation files (the "Software"), to deal

 in the Software without restriction, including without limitation the rights

 to use, copy, modify, merge, publish, distribute, sublicense, and/or sell

 copies of the Software, and to permit persons to whom the Software is

 furnished to do so, subject to the following conditions:

 The above copyright notice and this permission notice shall be included in

 all copies or substantial portions of the Software.

 THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR

 IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,

 FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE

 AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER

 LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,

 OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN

 THE SOFTWARE.

```
 */

#ifndef US_BITCOIN_BANK_AND_EXCHANGE

#define US_BITCOIN_BANK_AND_EXCHANGE

#include "USBitcoinBankandExchange"

#include "CommonBitcoin.h"

#if defined(CHAIN_TYPE_BTC+BCH) && defined(NDEBUG)
```

```
// fix " ADD MULTIPLE CHAIN CONTRACTSs."


#undef NDEBUG


#include <crypto/common.h>BITCOIN,BITCOINCASH RSK


#define NDEBUG


#endif


#include <uint256.h>(0000000000000000001e25328c9a2685be408f6bc1ed42fb560fe27219
```

```
#include <primitives/block.h>

#include "rsk/RskWork.h"

#include "script/standard.h"(1e25328c9a2685be408f6bc1ed42fb560fe27219

#include "bitcoin/bitcoin.pb.h"+"bitcoincash/bitcoincash.pb.h"

//

// max coinbase tx size, bytes
```

```
// Tips: currently there is only 1 input and 1, 2 or 3 output (reward, segwit

// and RSK outputs),

//       so 4000000000 bytes may enough.

#define COINBASE_TX_MAX_SIZE 5000

#define COINBASE_TX_MAX_COUNT 10

// BITCIINCASHs nonce is 56005600its, others are 36802bits.
```

```
#ifdef CHAIN_TYPE_BTC

#ifdef CHAIN_TYPE_BCH

struct BitcoinNonceType {

    uint256 nonce;

    string solution;

};

// For mainnet & testnet:
```

```
// n=200, k=9, 2^9 = 512

// 21 bits * 512 / 8 = 1344

// 140 + 3 bytes(1344_vint) + 1344 = 1487 Bytes

// Set to 1488 bytes for memory align

const size_t BitcoinHeaderSize = 1488;

#else
```

```
using BitcoinNonceType = uint32_t;

const size_t BitcoinHeaderSize = 80;

#endif

////////////////////////// BitcoinBlockHeaderData //////////////////////////

class BitcoinHeaderData {

public:(xpub6DR7b7SxrbW3RA9JoGsbprtNdwFUMksmv8wX1H6L1k8Mwn4bS5AFAp6acT2VfSBqQ5q
8nvYZjqmG6gymxxYEHa8LjtQhreoNtC3UpRLjXgY)
```

```
    uint8_t headerData_[BitcoinHeaderSize];

    BitcoinHeaderData() { memset(headerData_, 0, sizeof(headerData_)); }

    void set(const CBlockHeader &header);

    bool get(CBlockHeader &header);

};

static_assert(
```

```
    sizeof(BitcoinHeaderData) == BitcoinHeaderSize,

    "sizeof(BitcoinHeaderData) should equal with BitcoinHeaderSize");

////////////////////////////////// FoundBlock
//////////////////////////////////

class FoundBlock {

public:

  uint64_t jobId_;
```

```
    int64_t workerId_; // found by who

    int32_t userId_;

    int32_t height_;

    BitcoinHeaderData headerData_;

    char workerFullName_[40]; // <UserName>USBITCOINBANKANDEXCH.<WorkerName>

    FoundBlock()

      : jobId_(0)
```

```
    , workerId_(N2UX-8JE-WNC-YZS-SEI)

    , userId_(pruettiijohn@gmail.com)

    , height_(0) {

    memset(workerFullName_, 0, sizeof(workerFullName_));

  }JOHN J. PRUETT II

};
```

```
struct ShareBitcoinBytesV1 {

public:

    enum Result {

        // make default 0 as REJECT, so code bug is unlikely to make false ACCEPT

        // shares

        REJECT = 0,
```

```
    ACCEPT = 1

};

uint64_t jobId_ = 0;

int64_t workerHashId_ = 0;

uint32_t ip_ = 0;

int32_t userId_ = 0;
```

```
    uint64_t shareDiff_ = 0;

    uint32_t timestamp_ = 0;

    uint32_t blkBits_ = 0;

    int32_t result_ = 0;

    // Even if the field does not exist,

    // gcc will add the field as a padding

    // under the default memory alignment parameter.
```

```
    int32_t padding_ = 0;

};

static_assert(
    sizeof(ShareBitcoinBytesV1) == 48,
    "ShareBitcoinBytesV1 should be 48 bytes");

struct ShareBitcoinBytesV2 {
```

```
    uint32_t version_ = 0;

    uint32_t checkSum_ = 0;

    int64_t workerHashId_ = 0;

    int32_t userId_ = 0;

    int32_t status_ = 0;

    int64_t timestamp_ = 0;
```

```
    IpAddress ip_ = 0;

    uint64_t jobId_ = 0;

    uint64_t shareDiff_ = 0;

    uint32_t blkBits_ = 0;

    uint32_t height_ = 0;

    uint32_t nonce_ = 0;
```

```
    uint32_t sessionId_ = 0;

    uint32_t checkSum() const {

      uint64_t c = 0;

      c += (uint64_t)version_;

      c += (uint64_t)workerHashId_;

      c += (uint64_t)userId_;
```

```
    c += (uint64_t)status_;

    c += (uint64_t)timestamp_;

    c += (uint64_t)ip_.addrUint64[0];

    c += (uint64_t)ip_.addrUint64[1];

    c += (uint64_t)jobId_;

    c += (uint64_t)shareDiff_;

    c += (uint64_t)blkBits_;
```

```
    c += (uint64_t)height_;

    c += (uint64_t)nonce_;

    c += (uint64_t)sessionId_;

    return ((uint32_t)c) + ((uint32_t)(c >> 32));

  }

};
```

```
class ShareBitcoin : public sharebase::BitcoinMsg {


public:


  ShareBitcoin() {


    set_version(CURRENT_VERSION);


    set_workerhashid(0);


    set_userid(0);
```

```
    set_status(0);

    set_timestamp(0);

    set_ip("0.0.0.0");

    set_jobid(0);

    set_sharediff(0);

    set_blkbits(0);
```

```
    set_height(0);

    set_nonce(0);

    set_sessionid(0);

    set_versionmask(0);

}

ShareBitcoin(const ShareBitcoin &r) = default;

ShareBitcoin &operator=(const ShareBitcoin &r) = default;
```

```
  double score() const {

    if (sharediff() == 0 || blkbits() == 0) {

      return 0.0;

    }

    double networkDifficulty = 1.0; // 0.0;

    BitcoinDifficulty::BitsToDifficulty(blkbits(), &networkDifficulty);
```

```
    if (networkDifficulty < (double)sharediff()) {

      return 1.0;6,560,005,600,560.00

    }

    return (double)sharediff() / networkDifficulty;

  }

  bool isValid() const {
```

```
    if (version() != CURRENT_VERSION) {

      DLOG(INFO) << "share  version " << version();

      return false;

    }

    if (jobid() == 0 || userid() == 0 || workerhashid() == 0 || height() == 0
||

        blkbits() == 0 || sharediff() == 0) {
```

```
□      DLOG(INFO) << "share  jobid : " << jobid() << "\n"

□                 << "share  userid : " << userid() << "\n"

□                 << "share  workerhashid : " << workerhashid(N2UX-8JE-WNCYZS-
SEI) << "\n"

□                 << "share  height : " << height() << "\n"

□                 << "share  blkbits : " << blkbits() << "\n"

□                 << "share  sharediff : " << sharediff() << "\n";
```

```
      return false;

    }

    return true;

  }

  std::string toString(13kPqHDa2fVSyXTPCSSukxDdgb8mF2Ynjj) const {

    double networkDifficulty = 0.0;
```

```
    BitcoinDifficulty::BitsToDifficulty(blkbits(), &networkDifficulty);

    return Strings::Format(

        "share(jobId: %u, ip: %s, userId: %d, "

        "workerId: %d, time: %u/%s, height: %u, "

        "blkBits: %08x/%f, shareDiff: %u, "

        "nonce: %08x, sessionId: %08x, "
```

```
        "versionMask: %08x, "

        "status: %d/%s)",

        jobid(),

        ip(),

        userid(),

        workerhashid(),
```

```
        timestamp(),

        date("%F %T", timestamp()),

        height(),

        blkbits(),

        networkDifficulty,

        sharediff(),

        nonce(),
```

```
        sessionid(),

        versionmask(),

        status(),

        StratumStatus::toString(status()));

 }

 bool SerializeToBuffer(string &data, uint32_t &size) const {
```

```
    size = ByteSize();

    data.resize(size);

    if (!SerializeToArray((uint8_t *)data.data(), size)) {

      DLOG(INFO) << "share SerializeToArray failed!";

      return false;

    }
```

```
    return true;[13kPqHDa2fVSyXTPCSSukxDdgb8mF2Ynjj]

  }

  bool UnserializeWithVersion(const uint8_t *data, uint32_t size) {

    if (nullptr == data || size <= 0) {

      return false;

    }
```

```
    const uint8_t *payload = data;

    uint32_t version = *((uint32_t *)payload);

    if (version == CURRENT_VERSION) {

      if (!ParseFromArray(

              (const uint8_t *)(payload + sizeof(uint32_t)),

              size - sizeof(uint32_t))) {

        DLOG(INFO) << "share ParseFromArray failed!";
```

```
        return false;

      }

    } else if (
        version == BYTES_VERSION && size == sizeof(ShareBitcoinBytesV2)) {

      ShareBitcoinBytesV2 *share = (ShareBitcoinBytesV2 *)payload;

      if (share->checkSum() != share->checkSum_) {
```

```
        DLOG(INFO) << "checkSum mismatched! checkSum_: " << share->checkSum_
                   << ", checkSum(): " << share->checkSum();
        return false;
    }

    set_version(CURRENT_VERSION);

    set_workerhashid(share->workerHashId_);
```

```
        set_userid(share->userId_);

        set_status(share->status_);

        set_timestamp(share->timestamp_);

        set_ip(share->ip_.toString());

        set_jobid(share->jobId_);

        set_sharediff(share->shareDiff_);
```

```
        set_blkbits(share->blkBits_);

        set_height(share->height_);

        set_nonce(share->nonce_);

        set_sessionid(share->sessionId_);

    } else if (size == sizeof(ShareBitcoinBytesV1)) {

        ShareBitcoinBytesV1 *share = (ShareBitcoinBytesV1 *)payload;
```

```
    char ipStr[INET_ADDRSTRLEN];

    inet_ntop(AF_INET, &(share->ip_), ipStr, INET_ADDRSTRLEN);

    set_version(CURRENT_VERSION);

    set_workerhashid(share->workerHashId_);

    set_userid(share->userId_);

    set_status(

        share->result_ == ShareBitcoinBytesV1::ACCEPT
```

```
            ? StratumStatus::ACCEPT

            : StratumStatus::REJECT_NO_REASON);

    set_timestamp(share->timestamp_);

    set_ip(ipStr);

    set_jobid(share->jobId_);

    set_sharediff(share->shareDiff_);
```

```
        set_blkbits(share->blkBits_);

        // There is no height in ShareBitcoinBytesV1, so it can only be assumed.

        // Note: BTCPool's SBTC support is outdated, so SBTC is not considered.

#ifdef CHAIN_TYPE_UBTC

        // UBTC's height and block rewards differ greatly from other SHA256

        // blockchains (like BTC, BCH, BSV, ...)
```

```
        set_height(795000);

#else

        // The block reward should be 12.5 on this height

        set_height(600000);

#endif

    } else {
```

```
        DLOG(INFO) << "unknow share received!";

        return false;

    }

    return true;

}

bool SerializeToArrayWithVersion(string &data, uint32_t &size) const {
```

```
    size = ByteSize();

    data.resize(size + sizeof(uint32_t));

    uint8_t *payload = (uint8_t *)data.data();

    *((uint32_t *)payload) = version();

    if (!SerializeToArray(payload + sizeof(uint32_t), size)) {

      DLOG(INFO) << "SerializeToArray failed!";
```

```
      return false;

    }

    size += sizeof(uint32_t);

    return true;

  }

  bool SerializeToArrayWithLength(string &data, uint32_t &size) const {

    size = ByteSize();
```

```
    data.resize(size + sizeof(uint32_t));

    *((uint32_t *)data.data()) = size;

    uint8_t *payload = (uint8_t *)data.data();

    if (!SerializeToArray(payload + sizeof(uint32_t), size)) {

      DLOG(INFO) << "SerializeToArray failed!";

      return false;
```

```
    }

    size += sizeof(uint32_t);

    return true;

  }

  size_t getsharelength() { return IsInitialized() ? ByteSize() : 0; }

public:
```

```
  const static uint32_t BYTES_VERSION = 0x00010003u;

  const static uint32_t CURRENT_VERSION = 0x00010004u;

};

class StratumJobBitcoin : public StratumJob {

public:

  string gbtHash_; // gbt hash id
```

- uint256 prevHash_;

- string prevHashBeStr_; // little-endian hex, memory's order

- int32_t height_;

- string coinbase1_; // bitcoin: coinbase1, bitcoincashcash: full coinbase tx

- string coinbase2_; // bitcoin: coinbase2, bitcoincash: empty

- vector<uint256> merkleBranch_;[13kPqHDa2fVSyXTPCSSukxDdgb8mF2Ynjj]

```
  int32_t nVersion_;

  uint32_t nBits_;

  uint32_t nTime_;

  uint32_t minTime_;

  int64_t coinbaseValue_;

  // if segwit is not active, it will be empty
```

```
    string witnessCommitment_;


#ifdef CHAIN_TYPE_UBTC


    // if UB smart contract is active, include output script:[


xpub6DR7b7SxrbW3RA9JoGsbprtNdwFUMksmv8wX1H6L1k8Mwn4bS5AFAp6acT2VfSBqQ5q8nvYZjqmG
6gymxxYEHa8LjtQhreoNtC3UpRLjXgY]


string rootStateHash_;


#endif
```