Atlas

Cross-Platform C++ Bitcoin Wallet

Philip Glazman

Spring 2018

https://github.com/MangoSalad/AtlasWallet

Table of Contents

Introduction

Glossary

- Bitcoin – Network protocol used to reach consensus on who owns bitcoins.
- bitcoin – The value transferred in the Bitcoin protocol.
- Satoshis – The lowest denominator of bitcoin. One satoshi is 1/100millionth of a bitcoin.
- BIP – Bitcoin Improvement Proposals (BIP) are approved or pending proposals to the Bitcoin protocol. Several BIPs provide a standard for how the protocol or nodes should behave. This project uses several BIP standards regarding how wallets ought to be implemented.
- Mining – The process by which the network reaches consensus and a transaction is confirmed.
- Script – The programming language used by Bitcoin to write scripts. This language operates uses operation codes on a reverse polish notation stack.
- Smart Contract – A piece of code that is self-enforcing on the blockchain.
- Blockchain – Public data structure that maintains a ledger representing the entire state of the network.

For more information on these terms and others not covered, please consider reviewing the open-source Bitcoin wiki located here: https://en.bitcoin.it/

Background

Bitcoin is a digital money developed in 2009 where each node participating in the network can independently validate transactions and propagate them throughout the network using software similar to bittorrent. The protocol relies on public-key cryptography to create public addresses for the end-user. In terms of bitcoin, a wallet software manages the private keys that are associated with each public address. These keys gives users ownership in spending transactions and bitcoin. In bitcoin, the wallet is an abstraction that allows the end-user to send and receive payments.

Project

The aim of this project is to create a user-friendly bitcoin wallet implementation that encourages self-ownership of bitcoins and the use of bitcoin's Script language. Several Bitcoin wallets exist in the ecosystem but there does not exist a wallet that provides an abstraction layer that allows the end-user to interact with bitcoin smart contracts in a user-friendly way.

Atlas proposes a different way for the user to interact with how interact with Bitcoin. With a focus on financial independence through education, Atlas provides a straightforward way for the user to write smart contracts and learn more about the underlying low-level protocol.

Warning

This wallet was not extensively tested for security vulnerabilities, therefore should not be used with real bitcoin. The current implementation of Atlas operates on the Bitcoin test network and

uses test network bitcoins. Funds sent and received should be used with addresses that have a test network prefix.

Other Notes

This project heavily relied on Andreas M. Antonopoulos's *Mastering Bitcoin*, open-source documentation notes on Libbitcoin on the Libbitcoin Wiki, and Aaron Jaramillo's tutorials on Libbitcoin. These, among other scattered documentation along the web, were very helpful and resourceful. Several illustrations are used and referenced in this documentation that are from *Mastering Bitcoin*.

Installation Instructions

Before running Atlas, a couple of important libraries are needed on the local machine.

1. Boost

   Visit https://www.boost.org/users/download/

   ```
   $ brew install boost
   ```

2. Libbitcoin

   Visit https://github.com/libbitcoin/libbitcoin/tree/version3.

   ```
   $ ./autogen.sh
   $ ./configure
   ```

```
$ make
$ sudo make install
$ sudo ldconfig
```

More details on Libbitcoin installation can be found on Github README.md

3. Curl

   Visit https://curl.haxx.se/download.html

   ```
   $ brew install curl
   ```

4. JsonCPP

   Visit https://github.com/open-source-parsers/jsoncpp

   ```
   $ brew install jsoncpp
   ```

5. OpenSSL

   Visit https://www.openssl.org/

   ```
   $ brew install openssl
   ```

6. Run make file in /qt.

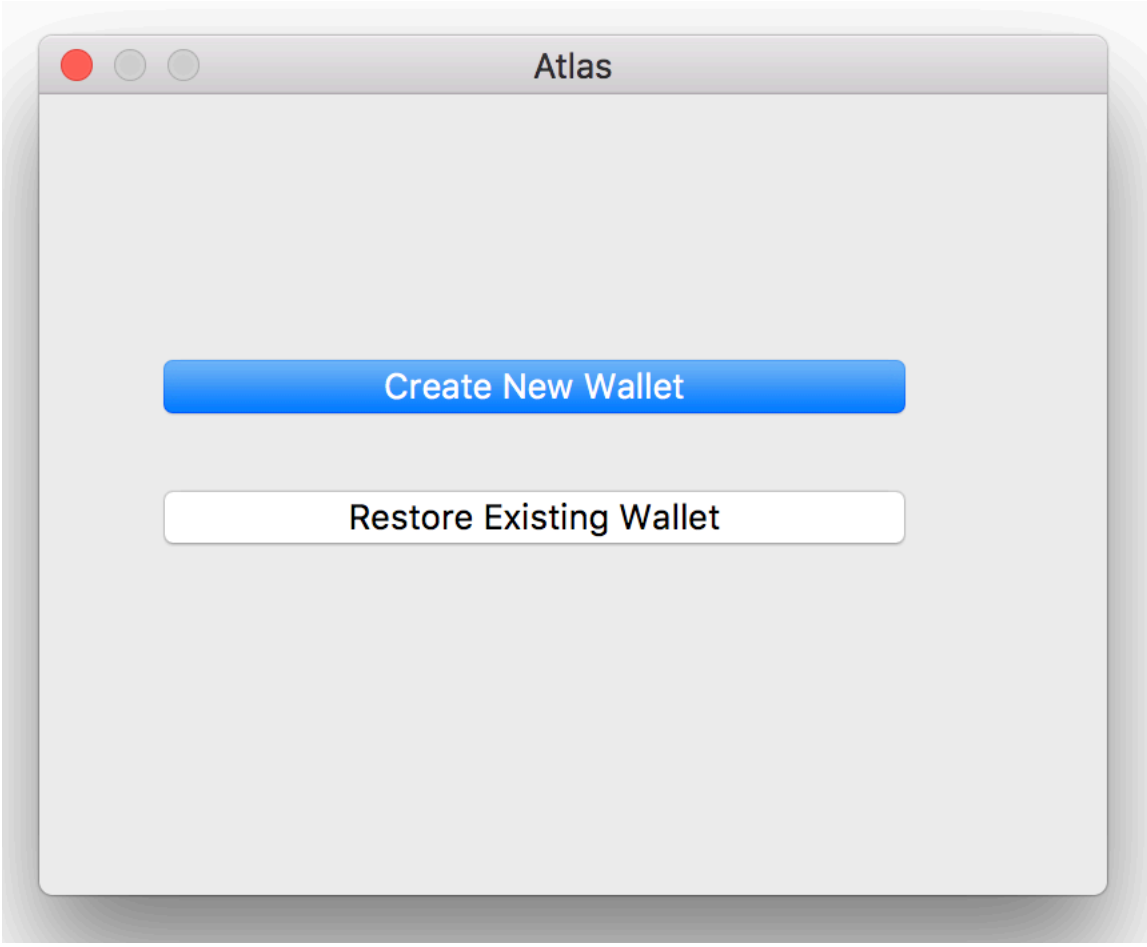
User Manual

Warning

Atlas uses industry standards for wallet management but there are several risks involved. As noted in the warning section in the Introduction, this wallet should not be used with real bitcoin.

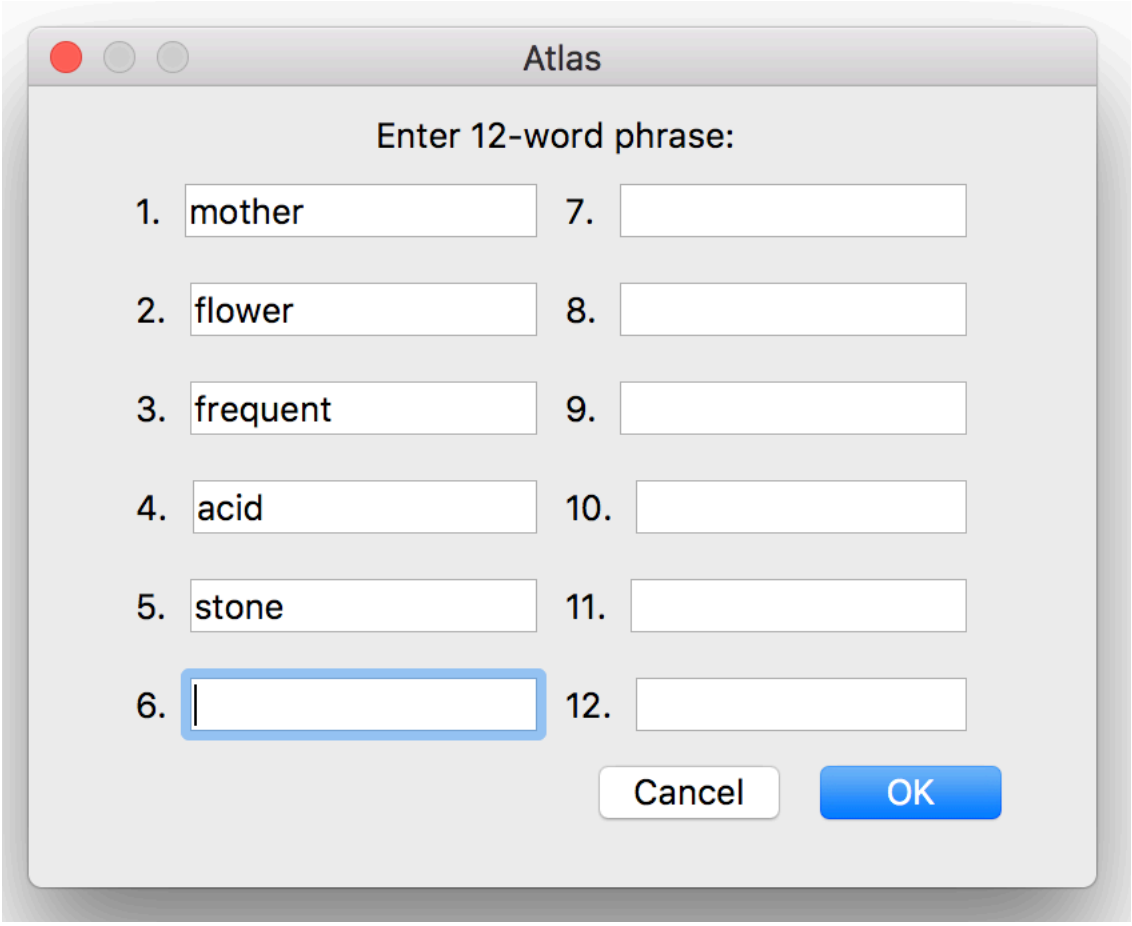
Wallet Creation

Upon starting the wallet, the user is prompted with a choice to either start a new wallet or restore an existing wallet. When a user starts a new wallet, a new seed is created that maintains the wallet. When a user chooses to restore a wallet, Atlas prompts the user for twelve words that comprises the mnemonic phrase. The user should keep these mnemonic words secret as they are the key to the wallet.



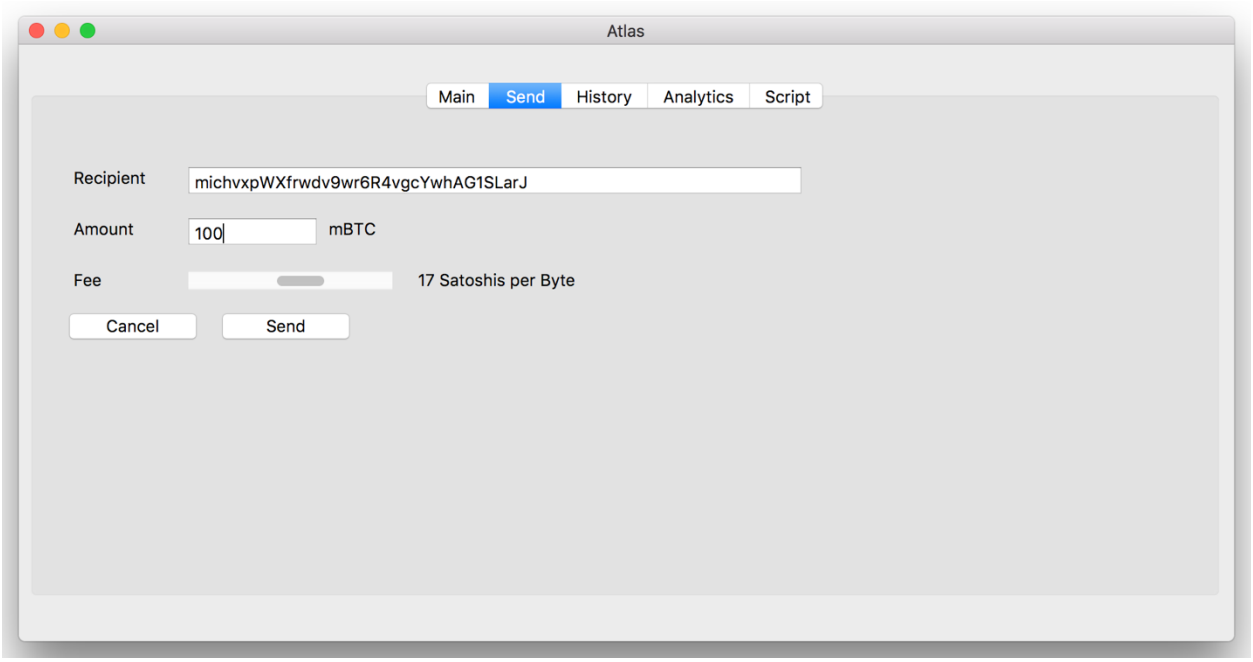
Mnemonic Phrase

The mnemonic phrase is a set of twelve words that create the wallet and make the wallet unique. This phrase should be kept secret for security. In its current implementation, the user cannot export their phrase. For development, a mnemonic phrase is included in the documentation in the Test Cases section. When starting the wallet, simply enter the 12 words into the boxes and it will restore an existing development wallet.



Sending Bitcoin

Sending a bitcoin transaction is very straightforward. A transaction in the Send tab allows the user to construct a basic transaction. A recipient represents another bitcoin payment address that the user will send funds to. The amount represents the number of bitcoins to send to the recipient. It is denominated in mBTC. The fees slider allows the user to change the amount of satoshis paid for the transaction fee. The user can have a minimum of zero satoshis per byte fees. It is important to note that Atlas sets a maximum fee which corresponds to the fee for the fastest transaction in the Analytics tab. This is implemented so that the user cannot overspend in fees.



Below is a chart of bitcoin denominations for reference. It is from the Bitcoin wiki.

Unit	Abbreviation	Decimal (BTC)	Alternate names	Info
Algorithmic maximum		20,999,999.9769		Calculation
tam-bitcoin		2,814,749.76710656		1,0000,0000 tonal
mega-bitcoin	MBTC	1,000,000		Rare in context
kilo-bitcoin	kBTC	1,000		Rare in context
hecto-bitcoin	hBTC	100		Rare
Initial block subsidy		50		Until block 210000[1]
bong-bitcoin	^{b}TBC	42.94967296		1,0000 tonal
Current block subsidy		12.5	block	As of block 420000
deca-bitcoin	daBTC	10		Rare
mill-bitcoin	^{m}TBC	2.68435456		1000 tonal
bitcoin	BTC	1	coin	SI base unit
san-bitcoin	^{s}TBC	0.16777216		100 tonal
deci-bitcoin	dBTC	0.1		Rare
ton-bitcoin	^{t}TBC	0.01048576		10 tonal
centi-bitcoin	cBTC	0.01	bitcent	Formerly frequent[2]
milli-bitcoin	mBTC	0.001	millibit, millie	Occasional
bitcoin	TBC	0.00065536		Tonal base unit
bitcoin-ton	TBC^{t}	0.00004096		0.1 tonal
bitcoin-san	TBC^{s}	0.00000256		0.01 tonal
micro-bitcoin	µBTC	0.000001	bit	Frequent
bitcoin-mill	TBC^{m}	0.00000016		0.001 tonal
		0.0000001	finney[3]	
bitcoin-bong	TBC^{b}	0.00000001		0.0001 tonal
	sat	0.00000001	satoshi	Blockchain value
	msat	0.00000000001	millisatoshi[4]	Payment channel value

Transaction History

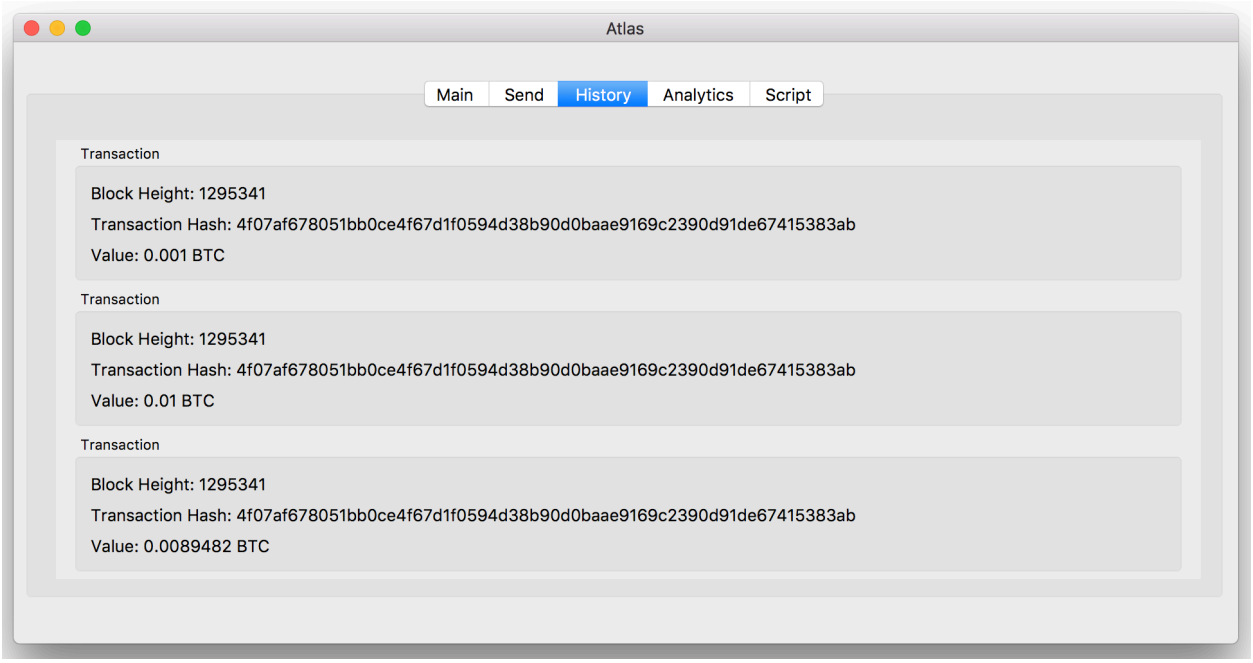
The History tab includes a list of transactions done by the addresses associated with the wallet.

Each box in scroll area includes a transaction with a block height, transaction hash, and value.

The block height represents where in the blockchain the transaction was confirmed, or mined.

The transaction hash is a unique identifier for the transaction that the user can later reference.

The value is the value of bitcoin transacted in that transaction.

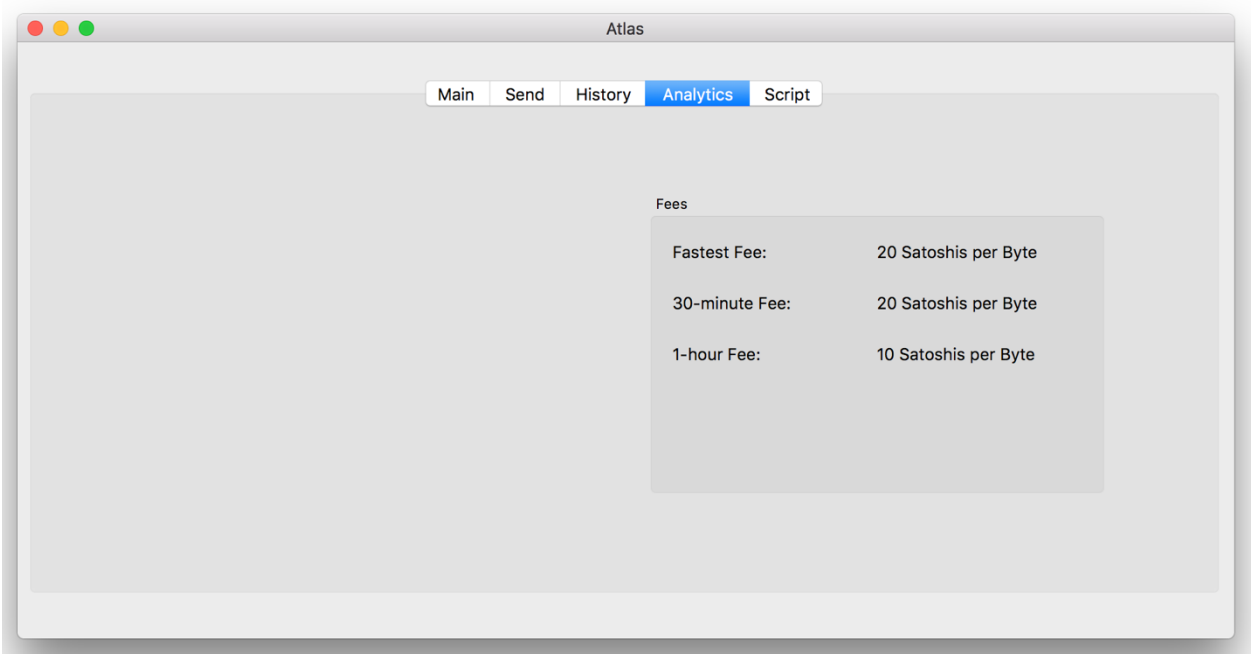


Understanding Network Fees

Transactions sent through the Bitcoin network may include a fee. The fee is optional and is set by the sender. A higher fee signals to the network that a specific transaction should be given priority for confirmation while a lower fee can lead to longer confirmation times. The fee market in Bitcoin is a free market set by supply and demand. As a result, the fee market changes over time and transactions can cost differently over the course of a day.

Atlas in its current implementation uses bitcoinfees.earn.com, a third-party API, to receive a suggested transaction fee. Using this API, Atlas is able to suggest to the user three fee costs located in the Analytics tab of Atlas. The user can choose to ignore these suggestions and selected a different fee.

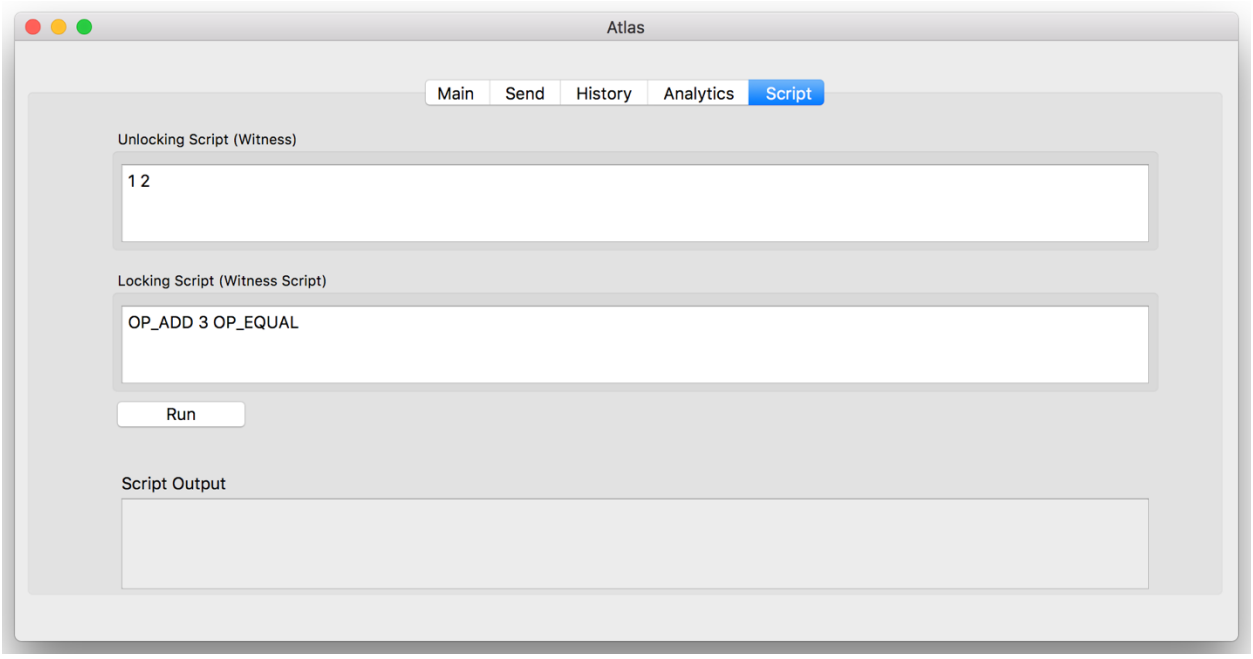
Fees are represented as Satoshi per Byte. This measurement informs the user that it costs n Satoshi per byte. If a transaction is 250 bytes in size and a suggested fee is 40 Satoshis per Byte, the fee will be 250*40 Satoshis.



Using Bitcoin Script

Bitcoin protocol uses a stack-based programming language called Script that allows the user to create smart contracts and develop on the Bitcoin protocol. The Script tab in Atlas provides a basic interface for the user to learn and engage with Bitcoin Script by validating scripts. Witness and Witness Script serve as text edits where the user can write a basic script.

In this example, 1 and 2 are pushed onto the stack. Operation code OP_ADD pops two items off the stack, adds them, and pushes the result onto the stack. Next, 3 is pushed onto the stack. Operation code OP_EQUAL evaluates that the two top items are equal. Given that they are equal in this case, a True Boolean is pushed onto the stack and the stack executes successfully. With only a True Boolean left on the stack, the script is valid.



Below is a list of available operation codes that can be used in the Atlas script interpreter. The definitions for each operation code is from Antonopoulos' *Mastering Bitcoin.*

Available Operation Codes	
OP_DROP	Pop the top item in the stack.
OP_DUP	Duplicate the top item in the stack.

OP_DEPTH	Count the items on the stack and push the resulting count.
OP_EQUAL	Push TRUE (1) if top two items are exactly equal, push FALSE (0) otherwise.
OP_1ADD	Add 1 to the top item.
OP_1SUB	Subtract 1 from the top item.
OP_NEGATE	Flip the sign of the top item.
OP_ABS	Change the sign of the top item to positive.
OP_ADD	Pop top two items, add them and push result.
OP_SUB	Pop top two items, subtract first from second, push result.
OP_NUMEQUAL	Return TRUE if top two items are equal numbers.
OP_NUMNOTEQUAL	Return TRUE if top two items are not equal numbers.
OP_LESSTHAN	Return TRUE if second item is less than top item.
OP_GREATERTHAN	Return TRUE if second item is greater than top item.

OP_LESSTHANOREQUAL	Return TRUE if second item is less than or equal to top item.
OP_GREATERTHANOREQUAL	Return TRUE if second item is greater than or equal to top item.
OP_MIN	Return the smaller item of the two top items.
OP_MAX	Return the larger of the two top items.
OP_WITHIN	Return TRUE if the third item is between the second item (or equal) and first item.
OP_RIPEMD160	Return RIPEMD160 hash of top item.
OP_SHA1	Return SHA1 hash of top item.
OP_SHA256	Return SHA256 hash of top item.
OP_HASH160	Return RIPEMD160(SHA256(x)) hash of top item.
OP_HASH256	Return SHA256(SHA256(x)) hash of top item.

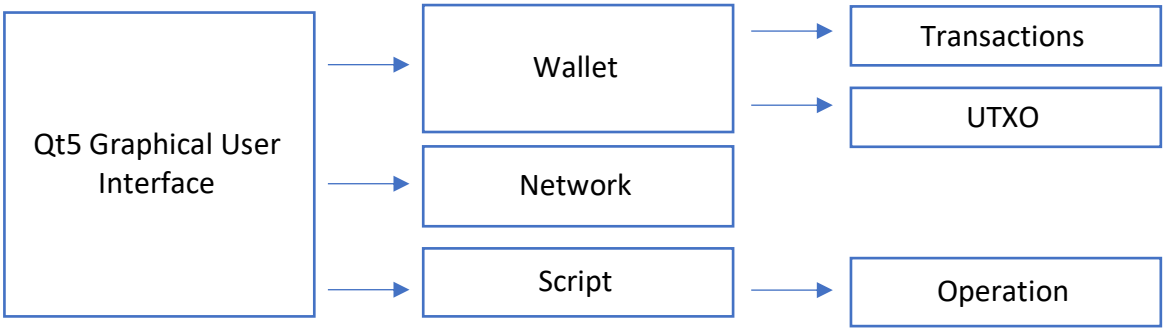
The script output section of this tab serves as a console for the user where it is printed if a script is valid or not.

Design

Philosophy

The wallet is designed in object-oriented principles. Objects are modularized and data is encapsulated in order to preserved object-oriented principles. The wallet is organized into basic wallet functionality, Bitcoin script manipulation, network functionality, utilities, and the front-end application.

Overview



Components

List of high-level components that make the wallet fundamentals.

Base Wallet

Components	**Description**
Mnemonic Code Words	Following a wallet standard, generated entropy will translate to 12 English words from a set. These words in addition to salt, will lead to a seed that creates a unique

	wallet. These 12 words could be written down and entered into the wallet to create this same unique wallet.
HD Wallet	A HD wallet, or deterministic wallet, is a wallet that creates a keychain based on a 512 bit seed. This is a standard in current Bitcoin wallets.
Bloom Filter	Bloom Filter is a standard privacy feature that allows the user to query for transactions without revealing to the network the specific transactions that he/she is asking for.
Peer Networking	Wallet connects to peers, does hand-shaking, and ask for transaction data. Most of the low-level work is handled by Libbitcoin library.
Payments	Allow user to send bitcoin and see the amount of bitcoin received. Allow user to generate new unique address for when receiving bitcoin
Graphical User Interface	Cross-platform Qt5 GUI.

Analytics

Components	Description
Fee Estimation	Query network to determine low, high, and median transaction fees. Provide recommendation to user for fee cost and when to send transaction.
Spend Analysis	Provide user with information on where bitcoin have been spent.

Script

Components	Description
Console	User can write their own bitcoin scripts and send them as transactions. Bitcoin Script language is stack-based language with limited OP codes.
Script debugger	Include debugger to help catch any errors in the user's script.

Classes

List of classes for backend and frontend of Atlas.

Network Classes

Network	• Provides network functionality to the wallet including broadcasting transactions, reading data from Bitcoin blockchain, and accessing the bitcoinfees.earn.com API for transaction fee recommendations. • Utility functions for accessing fee recommendations.
Bloomfilter	• Privacy feature for querying network for inputs in a transaction including unspent transaction outputs (utxo).

Script	
Script	• Provides functionality of Bitcoin script by simulating operations on a stack. • Includes a stack that serves as the execution stack for Bitcoin script.
Operation	• Includes operation codes and their functionality. Each operation code performs a function on the provided execution stack.

	• Includes utility cryptographic functions that can be used on the Bitcoin execution stack.

Utility	
Valid_Address	• Utility for validating if a string matches the consensus rules of the Bitcoin blockchain.

Wallet	
Wallet	• Creates wallet seed. • Responsible for key management. • Responsible for address management.
utxo	• Manages the wallet's record of unspent transaction outputs.
Transaction	• Provides utility functions for building a transaction. • Functionality to broadcast transaction when connected to the network. • Holds a history of previous transactions related to the wallet.

Error	• Handles exceptions and has an error stack. • Provides error log for debugging.

Qt	
App	• User interface for the main wallet application. • User interface is divided into four tabs.
Restore_wallet	• User interface for mnemonic prase input. • Validation of user input.
Start_menu	• User interface for selecting to start a new wallet or restore an existing wallet.

Data Structures

Important data structures that provides functionality for wallet fundamentals.

Data Structures

feeEstimation	• Struct stores three fee recommendations to the user. • Contains satoshi value for fees that will result in transaction confirmation in fastest time, 30 minutes, or 60 minutes.
m_ErrorMsgs	• Error stack that contains any errors that occurred within the wallet during runtime.
m_tx	• Tuple data type representing any spent and confirmed transaction. • Contains satoshi value of transaction, transaction hash, and block height of the confirmed transaction.
m_utxo	• Vector of tuples holding all unspent transaction outputs. These transactions outputs are spendable by the wallet. • Tuple contains satoshi value of transaction, transaction hash, and Bitcoin payment address.

op_code_map	• Hash map that maps a string representing an operation code to a function that manipulates the stack. • The hash map provides efficient lookup for the program to manipulate the Bitcoin execution stack with a given operation code.
m_execution_stack	• Execution stack that represents the Bitcoin execution stack. • The stack is manipulated by operation codes.

Insights and Challenges

UTXO Management

The management of each transaction under the hood was more difficult than planned. The function of a wallet is to create an efficient manner for organizing transactions. Each payment address might have several different transactions associated with it. Each transaction will have varying values of bitcoins. The aim of any wallet is to provide a way to send bitcoin from any previous transaction. This becomes complicated fairly quickly as organizing transactions has a

direct affect on the fees that the user pays. If an outgoing transaction relies on many input transactions, the fees paid by the user will be higher because the transaction size increases. Atlas uses a basic algorithm for sorting unspent transaction outputs on value. Going forward, a weighted approach will be needed so that value and number of inputs can be taken into account.

Transaction Building

Atlas currently only builds transactions that are pay-to-public-key-hash (P2PKH) which constitutes more than 80% of all transactions on the bitcoin network. The aim of Atlas was to experiment with building transactions that are more complex and less prevalent. There is still significant work to be done on transaction building as well as more efficient organization on how transactions are built.

Conclusion

Writing this program was intensive ultimately I am still not satisfied with the product and will have to continue to update the project. There are several features that must be added in order for the wallet to become reliable and be able to use real Bitcoin funds. In this section, I will review the overall opportunities gained in designing the wallet and what I plan to add to it.

Opportunities

The opportunities in education gained by building a bitcoin wallet are very meaningful. The Bitcoin protocol has been in uninterrupted operation for nine years and does not show signs in losing relevancy. The venture capital money, developer interest, and philosophical intrigue into Bitcoin makes it worthwhile for at least some brief interest for any computer scientist.

The motivation in building Atlas were found in a desire to better understand the Bitcoin protocol. Most Bitcoin users interact with the protocol through the wallet abstraction layer, therefore understanding the mechanics of this software can allow a developer to significantly improve the way in which people interact with the protocol. Atlas was designed as an educational product that I hope to soon develop into an industry level product. The exciting challenges in learning how transactions and protocol work has also invited new ideas to explore going forward.

On a technical level, designing Atlas has made me more comfortable in designing large programs as well as have a better understanding on how to design object-oriented code. Atlas is not a perfect example of object-oriented principles, but I now know the underlying issues in order to challenge the code base and make it near-perfect.

The C++ language for this project for its object-oriented design and flexibility of memory management. The language is cross-platform which allows for flexibility on the devices that can run it. The Libbitcoin bitcoin development library was very resourceful in designing the wallet

because it was able to abstract several low-level cryptography and functionality. Boost library was also resourceful in using property trees data structure. The C++ is time-tested, has significant developer resources, and is overall a very flexible language that gives the programmer a large amount of control over detail.

Using the Libbitcoin library API was very challenging initially due to the scarce resources available for its latest version. However, the documentation is growing and being updated more consistently since starting Atlas. Become comfortable with another large project library while developing my own large project was overall a uniquely satisfying educational experience.

Bitcoin's ecosystem is nascent, but there are several industry standards that were used in the design of Atlas. In particular, BIP 39 for mnemonic phrase and BIP 32 for the hierarchical design of the wallet's keychain. Designing Atlas incentivized me to understand industry standards and learn more about the Bitcoin ecosystem.

Designing Atlas required a large learning curve about Bitcoin and a basic understanding of its cryptography. Andreas Antonopoulos' *Mastering Bitcoin* was incredibly resourceful that provided a strong technical foundation in Bitcoin's protocol.

The graphical user interface of Atlas was written in Qt5 which is a cross-platform library for cross-platform applications. Qt5 has both extensive documentation and very reliable codebase for a free product. As part of using the library, it was agreed that Atlas will remain open-source.

Next Steps

There are several next steps to Atlas in order to improve the functionality and reliability of the application. In particular, BIP 21 will be implemented in order to allow QR codes to be presented. In addition, better fee recommendations and analytics will need to be added in order for the user to have a better understanding of their funds. In the initial proposal of the wallet, providing analytics was a major pillar of the application. Unfortunately, Atlas only provides fee estimation through a 3rd party API. Going forward, it must not rely on the API and instead use dynamic fee estimation. In addition to fee estimation, an analytics dashboard will be implemented that shows where and how bitcoin are spent.

There is still significant work to be done for the user's exposure to the Bitcoin script language. In its current form, the user can only test and debug a Bitcoin script. Going forward, the user should be provided a way to submit transactions with their written script. In addition, after conversations with helpful developers as well as reading supplementary material, there are new ways in developing a more user-friendly approach to script construction that I will have to implement.

Bibliography

The following resources were very helpful in building Atlas.

Antonopoulos, Andreas M. *Mastering Bitcoin: Programming the Open Blockchain.* 2nd ed., O'Reilly Media, 2017.

Jaramillo, Aaron. “The Libbitcoin Tutorials.” *The Web Log of Aaron Jaramillo*, aaronjaramillo.org/category/libbitcoindocs.

“Libbitcoin Wiki.” *GitHub, Github*, github.com/libbitcoin/libbitcoin/wiki.

“Libbitcoin Documentation.” *Overview - Libbitcoin 1 Documentation*, libbitcoin.dyne.org/doc/overview.html.

Source Code

```
~~~~~~~~~~~~~~~~~~~~Source code for file network.cpp~~~~~~~~~~~~~~~~~~~~
#include "../wallet/stdafx.h"


Network::Network()
{
   m_client = NULL;
   m_fees = new feeEstimation;
}

Network::~Network()
{
   delete m_client;
   delete m_fees;
}
```

```
/**
 * @brief Returns obelisk client pointer that allows rpc calls to be done.
 *
 * @return bc::client::obelisk_client&
 */
bc::client::obelisk_client& Network::connect()
{
   // Testnet connection details.
   bc::client::connection_type connection = {};
         connection.retries = 3;
         connection.timeout_seconds = 8;
         connection.server = bc::config::endpoint("tcp://testnet3.libbitcoin.net:19091");

   //TODO Timeouts?
   //List of servers: https://github.com/libbitcoin/libbitcoin-server/wiki/Community-Servers

   // Initialize obelisk.
   m_client = new bc::client::obelisk_client(connection);

   // Check if connection is working.
         if(m_client->connect(connection))
   {
      std::cout << "Connected to Libbitcoin.net" << std::endl;
      return *m_client;
         }
   else
   {
      Error::RecordError(std::string("Error connecting to bitcoin network."));
      // should probably return something else.
      return *m_client;
   }
};

bool Network::disconnect()
{
   std::cout << "Disconnected from Libbitcoin.net" << std::endl;
   delete m_client;
   m_client = NULL;
};

// from stackoverflow
std::size_t callback(const char* in, std::size_t size, std::size_t num, std::string* out)
{
```

```
    const std::size_t totalBytes(size * num);
    out->append(in, totalBytes);
    return totalBytes;
};


void Network::refreshFeeRecommendations()
{
    // Instantiate curl objects.
    CURL *curl;
    CURLcode res;
    std::string buffer;

    // Init curl.
    curl_global_init(CURL_GLOBAL_DEFAULT);
    curl = curl_easy_init();

    if(curl)
    {
        // Request fee recommendations from bitcoinfees.earn.com (trusted recommendation).
        // TODO: internal free recommendation tool.
        curl_easy_setopt(curl, CURLOPT_URL,
"https://bitcoinfees.earn.com/api/v1/fees/recommended");

        // Timeout after 10 seconds.
        curl_easy_setopt(curl, CURLOPT_TIMEOUT, 10);

        std::unique_ptr<std::string> httpData(new std::string());

        // Response.
        curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, callback);
        curl_easy_setopt(curl, CURLOPT_WRITEDATA, httpData.get());
        res = curl_easy_perform(curl);

        // Error checking.
        if(res != CURLE_OK)
        {
            fprintf(stderr, "curl_easy_perform() failed: %s\n", curl_easy_strerror(res));
        }
        else
        {
            Json::Value jsonData;
            Json::Reader jsonReader;
```

```
        // Parse JSON.
        if (jsonReader.parse(*httpData, jsonData))
        {
            std::cout << "Successfully parsed JSON data" << std::endl;
            std::cout << "\nJSON data received:" << std::endl;
            std::cout << jsonData.toStyledString() << std::endl;

            // Load fees into feeEstimation struct.
            m_fees -> fastestFee = jsonData["fastestFee"].asUInt64();
            m_fees -> halfHourFee = jsonData["halfHourFee"].asUInt64();
            m_fees -> hourFee = jsonData["hourFee"].asUInt64();
            std::cout<<m_fees<<std::endl;
        };
    };

    // Clean-up.
    curl_easy_cleanup(curl);
  };

  // Clean-up
  curl_global_cleanup();

};
```

~~~~~~~~~~~~~~~~~~~~Source code for file bloomfilter.cpp~~~~~~~~~~~~~~~~~~~~

```
#include "stdafx.h"
#include "BloomFilter.h"
#include "Error.h"

/**/
/*
BloomFilter::bloomFilterHash()
NAME
BloomFilter::bloomFilterHash()
SYNOPSIS
void BloomFilter::bloomFilterHash()
DESCRIPTION
Creates hash for bloom filter.
RETURNS
Returns murmur3 hash.
AUTHOR
Philip Glazman
DATE
```
~~~~~~~~~~~~~~~~~~~~

```
1/11/2018
*/
/**/
void BloomFilter::bloomFilterHash()
{
   //nHashNum * 0xFBA4C795 + nTweak
   int murmurSeed = 0xFBA4C795 + m_nTweak;
   //TODO import murmur3 from hash.h (Bitcoin/Bitcoin)
}


~~~~~~~~~~~~~~~~~~~~Source code for file ui_restore_wallet.h~~~~~~~~~~~~~~~~~~~~
/*****************************************************************************
***
** Form generated from reading UI file 'restore_wallet.ui'
**
** Created by: Qt User Interface Compiler version 5.10.1
**
** WARNING! All changes made in this file will be lost when recompiling UI file!
*****************************************************************************
**/

#ifndef UI_RESTORE_WALLET_H
#define UI_RESTORE_WALLET_H

#include <QtCore/QVariant>
#include <QtWidgets/QAction>
#include <QtWidgets/QApplication>
#include <QtWidgets/QButtonGroup>
#include <QtWidgets/QDialog>
#include <QtWidgets/QDialogButtonBox>
#include <QtWidgets/QHBoxLayout>
#include <QtWidgets/QHeaderView>
#include <QtWidgets/QLabel>
#include <QtWidgets/QLineEdit>
#include <QtWidgets/QVBoxLayout>
#include <QtWidgets/QWidget>

QT_BEGIN_NAMESPACE

class Ui_restore_wallet
{
public:
   QDialogButtonBox *buttonBox;
```

```
QLabel *label_13;
QWidget *widget;
QHBoxLayout *horizontalLayout_13;
QVBoxLayout *verticalLayout;
QHBoxLayout *horizontalLayout;
QLabel *label;
QLineEdit *word_1;
QHBoxLayout *horizontalLayout_2;
QLabel *label_2;
QLineEdit *word_2;
QHBoxLayout *horizontalLayout_3;
QLabel *label_3;
QLineEdit *word_3;
QHBoxLayout *horizontalLayout_4;
QLabel *label_4;
QLineEdit *word_4;
QHBoxLayout *horizontalLayout_5;
QLabel *label_5;
QLineEdit *word_5;
QHBoxLayout *horizontalLayout_6;
QLabel *label_6;
QLineEdit *word_6;
QVBoxLayout *verticalLayout_2;
QHBoxLayout *horizontalLayout_7;
QLabel *label_7;
QLineEdit *word_7;
QHBoxLayout *horizontalLayout_8;
QLabel *label_8;
QLineEdit *word_8;
QHBoxLayout *horizontalLayout_9;
QLabel *label_9;
QLineEdit *word_9;
QHBoxLayout *horizontalLayout_10;
QLabel *label_10;
QLineEdit *word_10;
QHBoxLayout *horizontalLayout_11;
QLabel *label_11;
QLineEdit *word_11;
QHBoxLayout *horizontalLayout_12;
QLabel *label_12;
QLineEdit *word_12;

void setupUi(QDialog *restore_wallet)
{
```

```
if (restore_wallet->objectName().isEmpty())
    restore_wallet->setObjectName(QStringLiteral("restore_wallet"));
restore_wallet->resize(400, 300);
buttonBox = new QDialogButtonBox(restore_wallet);
buttonBox->setObjectName(QStringLiteral("buttonBox"));
buttonBox->setGeometry(QRect(30, 250, 341, 32));
buttonBox->setOrientation(Qt::Horizontal);
buttonBox->setStandardButtons(QDialogButtonBox::Cancel|QDialogButtonBox::Ok);
buttonBox->setCenterButtons(false);
label_13 = new QLabel(restore_wallet);
label_13->setObjectName(QStringLiteral("label_13"));
label_13->setGeometry(QRect(130, 10, 141, 16));
widget = new QWidget(restore_wallet);
widget->setObjectName(QStringLiteral("widget"));
widget->setGeometry(QRect(40, 30, 312, 218));
widget->setAutoFillBackground(false);
widget->setInputMethodHints(Qt::ImhNone);
horizontalLayout_13 = new QHBoxLayout(widget);
horizontalLayout_13->setObjectName(QStringLiteral("horizontalLayout_13"));
horizontalLayout_13->setContentsMargins(0, 0, 0, 0);
verticalLayout = new QVBoxLayout();
verticalLayout->setObjectName(QStringLiteral("verticalLayout"));
horizontalLayout = new QHBoxLayout();
horizontalLayout->setObjectName(QStringLiteral("horizontalLayout"));
label = new QLabel(widget);
label->setObjectName(QStringLiteral("label"));
label->setAutoFillBackground(false);
label->setInputMethodHints(Qt::ImhNone);

horizontalLayout->addWidget(label);

word_1 = new QLineEdit(widget);
word_1->setObjectName(QStringLiteral("word_1"));
word_1->setAutoFillBackground(false);
word_1->setInputMethodHints(Qt::ImhNone);
word_1->setMaxLength(50);
word_1->setClearButtonEnabled(false);

horizontalLayout->addWidget(word_1);


verticalLayout->addLayout(horizontalLayout);

horizontalLayout_2 = new QHBoxLayout();
```

```
horizontalLayout_2->setObjectName(QStringLiteral("horizontalLayout_2"));
label_2 = new QLabel(widget);
label_2->setObjectName(QStringLiteral("label_2"));
label_2->setAutoFillBackground(false);
label_2->setInputMethodHints(Qt::ImhNone);

horizontalLayout_2->addWidget(label_2);

word_2 = new QLineEdit(widget);
word_2->setObjectName(QStringLiteral("word_2"));
word_2->setAutoFillBackground(false);
word_2->setInputMethodHints(Qt::ImhNone);
word_2->setMaxLength(50);
word_2->setClearButtonEnabled(false);

horizontalLayout_2->addWidget(word_2);


verticalLayout->addLayout(horizontalLayout_2);

horizontalLayout_3 = new QHBoxLayout();
horizontalLayout_3->setObjectName(QStringLiteral("horizontalLayout_3"));
label_3 = new QLabel(widget);
label_3->setObjectName(QStringLiteral("label_3"));
label_3->setAutoFillBackground(false);
label_3->setInputMethodHints(Qt::ImhNone);

horizontalLayout_3->addWidget(label_3);

word_3 = new QLineEdit(widget);
word_3->setObjectName(QStringLiteral("word_3"));
word_3->setAutoFillBackground(false);
word_3->setInputMethodHints(Qt::ImhNone);
word_3->setMaxLength(50);
word_3->setClearButtonEnabled(false);

horizontalLayout_3->addWidget(word_3);


verticalLayout->addLayout(horizontalLayout_3);

horizontalLayout_4 = new QHBoxLayout();
horizontalLayout_4->setObjectName(QStringLiteral("horizontalLayout_4"));
label_4 = new QLabel(widget);
```

```
label_4->setObjectName(QStringLiteral("label_4"));
label_4->setAutoFillBackground(false);
label_4->setInputMethodHints(Qt::ImhNone);

horizontalLayout_4->addWidget(label_4);

word_4 = new QLineEdit(widget);
word_4->setObjectName(QStringLiteral("word_4"));
word_4->setAutoFillBackground(false);
word_4->setInputMethodHints(Qt::ImhNone);
word_4->setMaxLength(50);
word_4->setClearButtonEnabled(false);

horizontalLayout_4->addWidget(word_4);


verticalLayout->addLayout(horizontalLayout_4);

horizontalLayout_5 = new QHBoxLayout();
horizontalLayout_5->setObjectName(QStringLiteral("horizontalLayout_5"));
label_5 = new QLabel(widget);
label_5->setObjectName(QStringLiteral("label_5"));
label_5->setAutoFillBackground(false);
label_5->setInputMethodHints(Qt::ImhNone);

horizontalLayout_5->addWidget(label_5);

word_5 = new QLineEdit(widget);
word_5->setObjectName(QStringLiteral("word_5"));
word_5->setAutoFillBackground(false);
word_5->setInputMethodHints(Qt::ImhNone);
word_5->setMaxLength(50);
word_5->setClearButtonEnabled(false);

horizontalLayout_5->addWidget(word_5);


verticalLayout->addLayout(horizontalLayout_5);

horizontalLayout_6 = new QHBoxLayout();
horizontalLayout_6->setObjectName(QStringLiteral("horizontalLayout_6"));
label_6 = new QLabel(widget);
label_6->setObjectName(QStringLiteral("label_6"));
label_6->setAutoFillBackground(false);
```

```
label_6->setInputMethodHints(Qt::ImhNone);

horizontalLayout_6->addWidget(label_6);

word_6 = new QLineEdit(widget);
word_6->setObjectName(QStringLiteral("word_6"));
word_6->setAutoFillBackground(false);
word_6->setInputMethodHints(Qt::ImhNone);
word_6->setMaxLength(50);
word_6->setClearButtonEnabled(false);

horizontalLayout_6->addWidget(word_6);


verticalLayout->addLayout(horizontalLayout_6);


horizontalLayout_13->addLayout(verticalLayout);

verticalLayout_2 = new QVBoxLayout();
verticalLayout_2->setObjectName(QStringLiteral("verticalLayout_2"));
horizontalLayout_7 = new QHBoxLayout();
horizontalLayout_7->setObjectName(QStringLiteral("horizontalLayout_7"));
label_7 = new QLabel(widget);
label_7->setObjectName(QStringLiteral("label_7"));
label_7->setAutoFillBackground(false);
label_7->setInputMethodHints(Qt::ImhNone);

horizontalLayout_7->addWidget(label_7);

word_7 = new QLineEdit(widget);
word_7->setObjectName(QStringLiteral("word_7"));
word_7->setAutoFillBackground(false);
word_7->setInputMethodHints(Qt::ImhNone);
word_7->setMaxLength(50);
word_7->setClearButtonEnabled(false);

horizontalLayout_7->addWidget(word_7);


verticalLayout_2->addLayout(horizontalLayout_7);

horizontalLayout_8 = new QHBoxLayout();
horizontalLayout_8->setObjectName(QStringLiteral("horizontalLayout_8"));
```

```
label_8 = new QLabel(widget);
label_8->setObjectName(QStringLiteral("label_8"));
label_8->setAutoFillBackground(false);
label_8->setInputMethodHints(Qt::ImhNone);

horizontalLayout_8->addWidget(label_8);

word_8 = new QLineEdit(widget);
word_8->setObjectName(QStringLiteral("word_8"));
word_8->setAutoFillBackground(false);
word_8->setInputMethodHints(Qt::ImhNone);
word_8->setMaxLength(50);
word_8->setClearButtonEnabled(false);

horizontalLayout_8->addWidget(word_8);


verticalLayout_2->addLayout(horizontalLayout_8);

horizontalLayout_9 = new QHBoxLayout();
horizontalLayout_9->setObjectName(QStringLiteral("horizontalLayout_9"));
label_9 = new QLabel(widget);
label_9->setObjectName(QStringLiteral("label_9"));
label_9->setAutoFillBackground(false);
label_9->setInputMethodHints(Qt::ImhNone);

horizontalLayout_9->addWidget(label_9);

word_9 = new QLineEdit(widget);
word_9->setObjectName(QStringLiteral("word_9"));
word_9->setAutoFillBackground(false);
word_9->setInputMethodHints(Qt::ImhNone);
word_9->setMaxLength(50);
word_9->setClearButtonEnabled(false);

horizontalLayout_9->addWidget(word_9);


verticalLayout_2->addLayout(horizontalLayout_9);

horizontalLayout_10 = new QHBoxLayout();
horizontalLayout_10->setObjectName(QStringLiteral("horizontalLayout_10"));
label_10 = new QLabel(widget);
label_10->setObjectName(QStringLiteral("label_10"));
```

```
label_10->setAutoFillBackground(false);
label_10->setInputMethodHints(Qt::ImhNone);

horizontalLayout_10->addWidget(label_10);

word_10 = new QLineEdit(widget);
word_10->setObjectName(QStringLiteral("word_10"));
word_10->setAutoFillBackground(false);
word_10->setInputMethodHints(Qt::ImhNone);
word_10->setMaxLength(50);
word_10->setClearButtonEnabled(false);

horizontalLayout_10->addWidget(word_10);


verticalLayout_2->addLayout(horizontalLayout_10);

horizontalLayout_11 = new QHBoxLayout();
horizontalLayout_11->setObjectName(QStringLiteral("horizontalLayout_11"));
label_11 = new QLabel(widget);
label_11->setObjectName(QStringLiteral("label_11"));
label_11->setAutoFillBackground(false);
label_11->setInputMethodHints(Qt::ImhNone);

horizontalLayout_11->addWidget(label_11);

word_11 = new QLineEdit(widget);
word_11->setObjectName(QStringLiteral("word_11"));
word_11->setAutoFillBackground(false);
word_11->setInputMethodHints(Qt::ImhNone);
word_11->setMaxLength(50);
word_11->setClearButtonEnabled(false);

horizontalLayout_11->addWidget(word_11);


verticalLayout_2->addLayout(horizontalLayout_11);

horizontalLayout_12 = new QHBoxLayout();
horizontalLayout_12->setObjectName(QStringLiteral("horizontalLayout_12"));
label_12 = new QLabel(widget);
label_12->setObjectName(QStringLiteral("label_12"));
label_12->setAutoFillBackground(false);
label_12->setInputMethodHints(Qt::ImhNone);
```

```
    horizontalLayout_12->addWidget(label_12);

    word_12 = new QLineEdit(widget);
    word_12->setObjectName(QStringLiteral("word_12"));
    word_12->setAutoFillBackground(false);
    word_12->setInputMethodHints(Qt::ImhNone);
    word_12->setMaxLength(50);
    word_12->setClearButtonEnabled(false);

    horizontalLayout_12->addWidget(word_12);


    verticalLayout_2->addLayout(horizontalLayout_12);


    horizontalLayout_13->addLayout(verticalLayout_2);


    retranslateUi(restore_wallet);
    QObject::connect(buttonBox, SIGNAL(accepted()), restore_wallet, SLOT(accept()));
    QObject::connect(buttonBox, SIGNAL(rejected()), restore_wallet, SLOT(reject()));

    QMetaObject::connectSlotsByName(restore_wallet);
  } // setupUi

  void retranslateUi(QDialog *restore_wallet)
  {
    restore_wallet->setWindowTitle(QApplication::translate("restore_wallet", "Dialog",
nullptr));
    label_13->setText(QApplication::translate("restore_wallet", "Enter 12-word phrase:",
nullptr));
    label->setText(QApplication::translate("restore_wallet", "1.", nullptr));
    word_1->setPlaceholderText(QString());
    label_2->setText(QApplication::translate("restore_wallet", "2.", nullptr));
    word_2->setPlaceholderText(QString());
    label_3->setText(QApplication::translate("restore_wallet", "3.", nullptr));
    word_3->setPlaceholderText(QString());
    label_4->setText(QApplication::translate("restore_wallet", "4.", nullptr));
    word_4->setPlaceholderText(QString());
    label_5->setText(QApplication::translate("restore_wallet", "5.", nullptr));
    word_5->setPlaceholderText(QString());
    label_6->setText(QApplication::translate("restore_wallet", "6.", nullptr));
    word_6->setPlaceholderText(QString());
```

```
        label_7->setText(QApplication::translate("restore_wallet", "7.", nullptr));
        word_7->setPlaceholderText(QString());
        label_8->setText(QApplication::translate("restore_wallet", "8.", nullptr));
        word_8->setPlaceholderText(QString());
        label_9->setText(QApplication::translate("restore_wallet", "9.", nullptr));
        word_9->setPlaceholderText(QString());
        label_10->setText(QApplication::translate("restore_wallet", "10.", nullptr));
        word_10->setPlaceholderText(QString());
        label_11->setText(QApplication::translate("restore_wallet", "11.", nullptr));
        word_11->setPlaceholderText(QString());
        label_12->setText(QApplication::translate("restore_wallet", "12.", nullptr));
        word_12->setPlaceholderText(QString());
    } // retranslateUi

};

namespace Ui {
    class restore_wallet: public Ui_restore_wallet {};
} // namespace Ui

QT_END_NAMESPACE

#endif // UI_RESTORE_WALLET_H
```

~~~~~~~~~~~~~~~~~~~~Source code for file ui_start_wallet.h~~~~~~~~~~~~~~~~~~~~

```
/********************************************************************************
***
** Form generated from reading UI file 'start_wallet.ui'
**
** Created by: Qt User Interface Compiler version 5.10.1
**
** WARNING! All changes made in this file will be lost when recompiling UI file!
********************************************************************************
**/

#ifndef UI_START_WALLET_H
#define UI_START_WALLET_H

#include <QtCore/QVariant>
#include <QtWidgets/QAction>
#include <QtWidgets/QApplication>
#include <QtWidgets/QButtonGroup>
#include <QtWidgets/QHeaderView>
```
~~~~~~~~~~~~~~~~~~~~

```
#include <QtWidgets/QPushButton>
#include <QtWidgets/QVBoxLayout>
#include <QtWidgets/QWidget>

QT_BEGIN_NAMESPACE

class Ui_start_wallet
{
public:
    QWidget *layoutWidget;
    QVBoxLayout *verticalLayout;
    QPushButton *pushButton;
    QPushButton *restore_existing_wallet;

    void setupUi(QWidget *start_wallet)
    {
        if (start_wallet->objectName().isEmpty())
            start_wallet->setObjectName(QStringLiteral("start_wallet"));
        start_wallet->resize(400, 300);
        layoutWidget = new QWidget(start_wallet);
        layoutWidget->setObjectName(QStringLiteral("layoutWidget"));
        layoutWidget->setGeometry(QRect(80, 80, 291, 111));
        verticalLayout = new QVBoxLayout(layoutWidget);
        verticalLayout->setObjectName(QStringLiteral("verticalLayout"));
        verticalLayout->setContentsMargins(0, 0, 0, 0);
        pushButton = new QPushButton(layoutWidget);
        pushButton->setObjectName(QStringLiteral("pushButton"));

        verticalLayout->addWidget(pushButton);

        restore_existing_wallet = new QPushButton(layoutWidget);
        restore_existing_wallet->setObjectName(QStringLiteral("restore_existing_wallet"));

        verticalLayout->addWidget(restore_existing_wallet);


        retranslateUi(start_wallet);

        QMetaObject::connectSlotsByName(start_wallet);
    } // setupUi

    void retranslateUi(QWidget *start_wallet)
    {
        start_wallet->setWindowTitle(QApplication::translate("start_wallet", "Form", nullptr));
```

```
        pushButton->setText(QApplication::translate("start_wallet", "Create New Wallet",
nullptr));
        restore_existing_wallet->setText(QApplication::translate("start_wallet", "Restore Existing
Wallet", nullptr));
    } // retranslateUi

};

namespace Ui {
    class start_wallet: public Ui_start_wallet {};
} // namespace Ui

QT_END_NAMESPACE

#endif // UI_START_WALLET_H
```

~~~~~~~~~~~~~~~~~~~~Source code for file moc_app.cpp~~~~~~~~~~~~~~~~~~~~

```
/*****************************************************************************
** Meta object code from reading C++ file 'app.h'
**
** Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)
**
** WARNING! All changes made in this file will be lost!
*****************************************************************************/

#include "../Atlas/app.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
#if !defined(Q_MOC_OUTPUT_REVISION)
#error "The header file 'app.h' doesn't include <QObject>."
#elif Q_MOC_OUTPUT_REVISION != 67
#error "This file was generated using the moc from 5.10.1. It"
#error "cannot be used with the include files from this version of Qt."
#error "(The moc has changed too much.)"
#endif

QT_BEGIN_MOC_NAMESPACE
QT_WARNING_PUSH
QT_WARNING_DISABLE_DEPRECATED
struct qt_meta_stringdata_app_t {
    QByteArrayData data[3];
    char stringdata0[40];
};
```
~~~~~~~~~~~~~~~~~~~~

```
#define QT_MOC_LITERAL(idx, ofs, len) \
    Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \
    qptrdiff(offsetof(qt_meta_stringdata_app_t, stringdata0) + ofs \
        - idx * sizeof(QByteArrayData)) \
    )
static const qt_meta_stringdata_app_t qt_meta_stringdata_app = {
    {
QT_MOC_LITERAL(0, 0, 3), // "app"
QT_MOC_LITERAL(1, 4, 34), // "on_restore_existing_wallet_cl..."
QT_MOC_LITERAL(2, 39, 0) // ""

    },
    "app\0on_restore_existing_wallet_clicked\0"
    ""
};
#undef QT_MOC_LITERAL

static const uint qt_meta_data_app[] = {

 // content:
       7,       // revision
       0,       // classname
       0,    0, // classinfo
       1,   14, // methods
       0,    0, // properties
       0,    0, // enums/sets
       0,    0, // constructors
       0,       // flags
       0,       // signalCount

 // slots: name, argc, parameters, tag, flags
       1,    0,   19,    2, 0x08 /* Private */,

 // slots: parameters
    QMetaType::Void,

       0        // eod
};

void app::qt_static_metacall(QObject *_o, QMetaObject::Call _c, int _id, void **_a)
{
    if (_c == QMetaObject::InvokeMetaMethod) {
        app *_t = static_cast<app *>(_o);
        Q_UNUSED(_t)
```

```
        switch (_id) {
        case 0: _t->on_restore_existing_wallet_clicked(); break;
        default: ;
        }
    }
    Q_UNUSED(_a);
}

QT_INIT_METAOBJECT const QMetaObject app::staticMetaObject = {
    { &QMainWindow::staticMetaObject, qt_meta_stringdata_app.data,
      qt_meta_data_app,  qt_static_metacall, nullptr, nullptr}
};


const QMetaObject *app::metaObject() const
{
    return QObject::d_ptr->metaObject ? QObject::d_ptr->dynamicMetaObject() :
&staticMetaObject;
}

void *app::qt_metacast(const char *_clname)
{
    if (!_clname) return nullptr;
    if (!strcmp(_clname, qt_meta_stringdata_app.stringdata0))
        return static_cast<void*>(this);
    return QMainWindow::qt_metacast(_clname);
}

int app::qt_metacall(QMetaObject::Call _c, int _id, void **_a)
{
    _id = QMainWindow::qt_metacall(_c, _id, _a);
    if (_id < 0)
        return _id;
    if (_c == QMetaObject::InvokeMetaMethod) {
        if (_id < 1)
            qt_static_metacall(this, _c, _id, _a);
        _id -= 1;
    } else if (_c == QMetaObject::RegisterMethodArgumentMetaType) {
        if (_id < 1)
            *reinterpret_cast<int*>(_a[0]) = -1;
        _id -= 1;
    }
    return _id;
}
```

```
QT_WARNING_POP
QT_END_MOC_NAMESPACE


~~~~~~~~~~~~~~~~~~~~Source code for file ui_start_menu.h~~~~~~~~~~~~~~~~~~~~
/*****************************************************************************
***
** Form generated from reading UI file 'start_menu.ui'
**
** Created by: Qt User Interface Compiler version 5.10.1
**
** WARNING! All changes made in this file will be lost when recompiling UI file!
*****************************************************************************
**/

#ifndef UI_START_MENU_H
#define UI_START_MENU_H

#include <QtCore/QVariant>
#include <QtWidgets/QAction>
#include <QtWidgets/QApplication>
#include <QtWidgets/QButtonGroup>
#include <QtWidgets/QDialog>
#include <QtWidgets/QHeaderView>
#include <QtWidgets/QPushButton>
#include <QtWidgets/QVBoxLayout>
#include <QtWidgets/QWidget>

QT_BEGIN_NAMESPACE

class Ui_start_menu
{
public:
   QWidget *layoutWidget;
   QVBoxLayout *verticalLayout;
   QPushButton *create_new_wallet;
   QPushButton *restore_existing_wallet;

   void setupUi(QDialog *start_menu)
   {
      if (start_menu->objectName().isEmpty())
         start_menu->setObjectName(QStringLiteral("start_menu"));
      start_menu->resize(400, 300);
      layoutWidget = new QWidget(start_menu);
```

```
        layoutWidget->setObjectName(QStringLiteral("layoutWidget"));
        layoutWidget->setGeometry(QRect(40, 80, 291, 111));
        verticalLayout = new QVBoxLayout(layoutWidget);
        verticalLayout->setObjectName(QStringLiteral("verticalLayout"));
        verticalLayout->setContentsMargins(0, 0, 0, 0);
        create_new_wallet = new QPushButton(layoutWidget);
        create_new_wallet->setObjectName(QStringLiteral("create_new_wallet"));

        verticalLayout->addWidget(create_new_wallet);

        restore_existing_wallet = new QPushButton(layoutWidget);
        restore_existing_wallet->setObjectName(QStringLiteral("restore_existing_wallet"));

        verticalLayout->addWidget(restore_existing_wallet);


        retranslateUi(start_menu);

        QMetaObject::connectSlotsByName(start_menu);
    } // setupUi

    void retranslateUi(QDialog *start_menu)
    {
        start_menu->setWindowTitle(QApplication::translate("start_menu", "Dialog", nullptr));
        create_new_wallet->setText(QApplication::translate("start_menu", "Create New Wallet",
nullptr));
        restore_existing_wallet->setText(QApplication::translate("start_menu", "Restore Existing
Wallet", nullptr));
    } // retranslateUi

};

namespace Ui {
    class start_menu: public Ui_start_menu {};
} // namespace Ui

QT_END_NAMESPACE

#endif // UI_START_MENU_H
```

~~~~~~~~~~~~~~~~~~~~Source code for file moc_start_menu.cpp~~~~~~~~~~~~~~~~~~~~

```
/****************************************************************************
** Meta object code from reading C++ file 'start_menu.h'
```
~~~~~~~~~~~~~~~~~~~~

```
**
** Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)
**
** WARNING! All changes made in this file will be lost!
*****************************************************************************/

#include "../Atlas/start_menu.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
#if !defined(Q_MOC_OUTPUT_REVISION)
#error "The header file 'start_menu.h' doesn't include <QObject>."
#elif Q_MOC_OUTPUT_REVISION != 67
#error "This file was generated using the moc from 5.10.1. It"
#error "cannot be used with the include files from this version of Qt."
#error "(The moc has changed too much.)"
#endif

QT_BEGIN_MOC_NAMESPACE
QT_WARNING_PUSH
QT_WARNING_DISABLE_DEPRECATED
struct qt_meta_stringdata_start_menu_t {
    QByteArrayData data[4];
    char stringdata0[76];
};
#define QT_MOC_LITERAL(idx, ofs, len) \
    Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \
    qptrdiff(offsetof(qt_meta_stringdata_start_menu_t, stringdata0) + ofs \
        - idx * sizeof(QByteArrayData)) \
    )
static const qt_meta_stringdata_start_menu_t qt_meta_stringdata_start_menu = {
    {
QT_MOC_LITERAL(0, 0, 10), // "start_menu"
QT_MOC_LITERAL(1, 11, 28), // "on_create_new_wallet_clicked"
QT_MOC_LITERAL(2, 40, 0), // ""
QT_MOC_LITERAL(3, 41, 34) // "on_restore_existing_wallet_cl..."

    },
    "start_menu\0on_create_new_wallet_clicked\0"
    "\0on_restore_existing_wallet_clicked"
};
#undef QT_MOC_LITERAL

static const uint qt_meta_data_start_menu[] = {
```

```
 // content:
       7,       // revision
       0,       // classname
       0,    0, // classinfo
       2,   14, // methods
       0,    0, // properties
       0,    0, // enums/sets
       0,    0, // constructors
       0,       // flags
       0,       // signalCount

 // slots: name, argc, parameters, tag, flags
       1,    0,   24,    2, 0x08 /* Private */,
       3,    0,   25,    2, 0x08 /* Private */,

 // slots: parameters
    QMetaType::Void,
    QMetaType::Void,

       0        // eod
};

void start_menu::qt_static_metacall(QObject *_o, QMetaObject::Call _c, int _id, void **_a)
{
    if (_c == QMetaObject::InvokeMetaMethod) {
        start_menu *_t = static_cast<start_menu *>(_o);
        Q_UNUSED(_t)
        switch (_id) {
        case 0: _t->on_create_new_wallet_clicked(); break;
        case 1: _t->on_restore_existing_wallet_clicked(); break;
        default: ;
        }
    }
    Q_UNUSED(_a);
}

QT_INIT_METAOBJECT const QMetaObject start_menu::staticMetaObject = {
    { &QDialog::staticMetaObject, qt_meta_stringdata_start_menu.data,
      qt_meta_data_start_menu,  qt_static_metacall, nullptr, nullptr}
};


const QMetaObject *start_menu::metaObject() const
{
```

```
    return QObject::d_ptr->metaObject ? QObject::d_ptr->dynamicMetaObject() :
&staticMetaObject;
}

void *start_menu::qt_metacast(const char *_clname)
{
    if (!_clname) return nullptr;
    if (!strcmp(_clname, qt_meta_stringdata_start_menu.stringdata0))
        return static_cast<void*>(this);
    return QDialog::qt_metacast(_clname);
}

int start_menu::qt_metacall(QMetaObject::Call _c, int _id, void **_a)
{
    _id = QDialog::qt_metacall(_c, _id, _a);
    if (_id < 0)
        return _id;
    if (_c == QMetaObject::InvokeMetaMethod) {
        if (_id < 2)
            qt_static_metacall(this, _c, _id, _a);
        _id -= 2;
    } else if (_c == QMetaObject::RegisterMethodArgumentMetaType) {
        if (_id < 2)
            *reinterpret_cast<int*>(_a[0]) = -1;
        _id -= 2;
    }
    return _id;
}
QT_WARNING_POP
QT_END_MOC_NAMESPACE
```

~~~~~~~~~~~~~~~~~~~~Source code for file ui_app.h~~~~~~~~~~~~~~~~~~~~

```
/*********************************************************************
***
** Form generated from reading UI file 'app.ui'
**
** Created by: Qt User Interface Compiler version 5.10.1
**
** WARNING! All changes made in this file will be lost when recompiling UI file!
*********************************************************************
**/

#ifndef UI_APP_H
```
~~~~~~~~~~~~~~~~~~~~

```
#define UI_APP_H

#include <QtCore/QVariant>
#include <QtWidgets/QAction>
#include <QtWidgets/QApplication>
#include <QtWidgets/QButtonGroup>
#include <QtWidgets/QHeaderView>
#include <QtWidgets/QLabel>
#include <QtWidgets/QMainWindow>
#include <QtWidgets/QMenuBar>
#include <QtWidgets/QStatusBar>
#include <QtWidgets/QToolBar>
#include <QtWidgets/QWidget>

QT_BEGIN_NAMESPACE

class Ui_app
{
public:
    QWidget *centralWidget;
    QLabel *debuggerLabel;
    QMenuBar *menuBar;
    QToolBar *mainToolBar;
    QStatusBar *statusBar;

    void setupUi(QMainWindow *app)
    {
        if (app->objectName().isEmpty())
            app->setObjectName(QStringLiteral("app"));
        app->resize(953, 467);
        centralWidget = new QWidget(app);
        centralWidget->setObjectName(QStringLiteral("centralWidget"));
        debuggerLabel = new QLabel(centralWidget);
        debuggerLabel->setObjectName(QStringLiteral("debuggerLabel"));
        debuggerLabel->setGeometry(QRect(160, 180, 60, 16));
        app->setCentralWidget(centralWidget);
        menuBar = new QMenuBar(app);
        menuBar->setObjectName(QStringLiteral("menuBar"));
        menuBar->setGeometry(QRect(0, 0, 953, 22));
        app->setMenuBar(menuBar);
        mainToolBar = new QToolBar(app);
        mainToolBar->setObjectName(QStringLiteral("mainToolBar"));
        app->addToolBar(Qt::TopToolBarArea, mainToolBar);
        statusBar = new QStatusBar(app);
```

```
        statusBar->setObjectName(QStringLiteral("statusBar"));
        app->setStatusBar(statusBar);

        retranslateUi(app);

        QMetaObject::connectSlotsByName(app);
    } // setupUi

    void retranslateUi(QMainWindow *app)
    {
        app->setWindowTitle(QApplication::translate("app", "app", nullptr));
        debuggerLabel->setText(QApplication::translate("app", "TextLabel", nullptr));
    } // retranslateUi

};

namespace Ui {
    class app: public Ui_app {};
} // namespace Ui

QT_END_NAMESPACE

#endif // UI_APP_H
```

~~~~~~~~~~~~~~~~~~~~~Source code for file moc_predefs.h~~~~~~~~~~~~~~~~~~~~~

```
#define OBJC_NEW_PROPERTIES 1
#define _LP64 1
#define __APPLE_CC__ 6000
#define __APPLE__ 1
#define __ATOMIC_ACQUIRE 2
#define __ATOMIC_ACQ_REL 4
#define __ATOMIC_CONSUME 1
#define __ATOMIC_RELAXED 0
#define __ATOMIC_RELEASE 3
#define __ATOMIC_SEQ_CST 5
#define __BIGGEST_ALIGNMENT__ 16
#define __BLOCKS__ 1
#define __BYTE_ORDER__ __ORDER_LITTLE_ENDIAN__
#define __CHAR16_TYPE__ unsigned short
#define __CHAR32_TYPE__ unsigned int
#define __CHAR_BIT__ 8
#define __CONSTANT_CFSTRINGS__ 1
#define __DBL_DECIMAL_DIG__ 17
```
~~~~~~~~~~~~~~~~~~~~~

```
#define __DBL_DENORM_MIN__ 4.9406564584124654e-324
#define __DBL_DIG__ 15
#define __DBL_EPSILON__ 2.2204460492503131e-16
#define __DBL_HAS_DENORM__ 1
#define __DBL_HAS_INFINITY__ 1
#define __DBL_HAS_QUIET_NAN__ 1
#define __DBL_MANT_DIG__ 53
#define __DBL_MAX_10_EXP__ 308
#define __DBL_MAX_EXP__ 1024
#define __DBL_MAX__ 1.7976931348623157e+308
#define __DBL_MIN_10_EXP__ (-307)
#define __DBL_MIN_EXP__ (-1021)
#define __DBL_MIN__ 2.2250738585072014e-308
#define __DECIMAL_DIG__ __LDBL_DECIMAL_DIG__
#define __DEPRECATED 1
#define __DYNAMIC__ 1
#define __ENVIRONMENT_MAC_OS_X_VERSION_MIN_REQUIRED__ 101000
#define __EXCEPTIONS 1
#define __FINITE_MATH_ONLY__ 0
#define __FLT_DECIMAL_DIG__ 9
#define __FLT_DENORM_MIN__ 1.40129846e-45F
#define __FLT_DIG__ 6
#define __FLT_EPSILON__ 1.19209290e-7F
#define __FLT_EVAL_METHOD__ 0
#define __FLT_HAS_DENORM__ 1
#define __FLT_HAS_INFINITY__ 1
#define __FLT_HAS_QUIET_NAN__ 1
#define __FLT_MANT_DIG__ 24
#define __FLT_MAX_10_EXP__ 38
#define __FLT_MAX_EXP__ 128
#define __FLT_MAX__ 3.40282347e+38F
#define __FLT_MIN_10_EXP__ (-37)
#define __FLT_MIN_EXP__ (-125)
#define __FLT_MIN__ 1.17549435e-38F
#define __FLT_RADIX__ 2
#define __FXSR__ 1
#define __GCC_ATOMIC_BOOL_LOCK_FREE 2
#define __GCC_ATOMIC_CHAR16_T_LOCK_FREE 2
#define __GCC_ATOMIC_CHAR32_T_LOCK_FREE 2
#define __GCC_ATOMIC_CHAR_LOCK_FREE 2
#define __GCC_ATOMIC_INT_LOCK_FREE 2
#define __GCC_ATOMIC_LLONG_LOCK_FREE 2
#define __GCC_ATOMIC_LONG_LOCK_FREE 2
#define __GCC_ATOMIC_POINTER_LOCK_FREE 2
```

```
#define __GCC_ATOMIC_SHORT_LOCK_FREE 2
#define __GCC_ATOMIC_TEST_AND_SET_TRUEVAL 1
#define __GCC_ATOMIC_WCHAR_T_LOCK_FREE 2
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_1 1
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_16 1
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_2 1
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_4 1
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_8 1
#define __GLIBCXX_BITSIZE_INT_N_0 128
#define __GLIBCXX_TYPE_INT_N_0 __int128
#define __GNUC_GNU_INLINE__ 1
#define __GNUC_MINOR__ 2
#define __GNUC_PATCHLEVEL__ 1
#define __GNUC__ 4
#define __GNUG__ 4
#define __GXX_ABI_VERSION 1002
#define __GXX_EXPERIMENTAL_CXX0X__ 1
#define __GXX_RTTI 1
#define __GXX_WEAK__ 1
#define __INT16_C_SUFFIX__
#define __INT16_FMTd__ "hd"
#define __INT16_FMTi__ "hi"
#define __INT16_MAX__ 32767
#define __INT16_TYPE__ short
#define __INT32_C_SUFFIX__
#define __INT32_FMTd__ "d"
#define __INT32_FMTi__ "i"
#define __INT32_MAX__ 2147483647
#define __INT32_TYPE__ int
#define __INT64_C_SUFFIX__ LL
#define __INT64_FMTd__ "lld"
#define __INT64_FMTi__ "lli"
#define __INT64_MAX__ 9223372036854775807LL
#define __INT64_TYPE__ long long int
#define __INT8_C_SUFFIX__
#define __INT8_FMTd__ "hhd"
#define __INT8_FMTi__ "hhi"
#define __INT8_MAX__ 127
#define __INT8_TYPE__ signed char
#define __INTMAX_C_SUFFIX__ L
#define __INTMAX_FMTd__ "ld"
#define __INTMAX_FMTi__ "li"
#define __INTMAX_MAX__ 9223372036854775807L
#define __INTMAX_TYPE__ long int
```

```
#define __INTMAX_WIDTH__ 64
#define __INTPTR_FMTd__ "ld"
#define __INTPTR_FMTi__ "li"
#define __INTPTR_MAX__ 9223372036854775807L
#define __INTPTR_TYPE__ long int
#define __INTPTR_WIDTH__ 64
#define __INT_FAST16_FMTd__ "hd"
#define __INT_FAST16_FMTi__ "hi"
#define __INT_FAST16_MAX__ 32767
#define __INT_FAST16_TYPE__ short
#define __INT_FAST32_FMTd__ "d"
#define __INT_FAST32_FMTi__ "i"
#define __INT_FAST32_MAX__ 2147483647
#define __INT_FAST32_TYPE__ int
#define __INT_FAST64_FMTd__ "ld"
#define __INT_FAST64_FMTi__ "li"
#define __INT_FAST64_MAX__ 9223372036854775807L
#define __INT_FAST64_TYPE__ long int
#define __INT_FAST8_FMTd__ "hhd"
#define __INT_FAST8_FMTi__ "hhi"
#define __INT_FAST8_MAX__ 127
#define __INT_FAST8_TYPE__ signed char
#define __INT_LEAST16_FMTd__ "hd"
#define __INT_LEAST16_FMTi__ "hi"
#define __INT_LEAST16_MAX__ 32767
#define __INT_LEAST16_TYPE__ short
#define __INT_LEAST32_FMTd__ "d"
#define __INT_LEAST32_FMTi__ "i"
#define __INT_LEAST32_MAX__ 2147483647
#define __INT_LEAST32_TYPE__ int
#define __INT_LEAST64_FMTd__ "ld"
#define __INT_LEAST64_FMTi__ "li"
#define __INT_LEAST64_MAX__ 9223372036854775807L
#define __INT_LEAST64_TYPE__ long int
#define __INT_LEAST8_FMTd__ "hhd"
#define __INT_LEAST8_FMTi__ "hhi"
#define __INT_LEAST8_MAX__ 127
#define __INT_LEAST8_TYPE__ signed char
#define __INT_MAX__ 2147483647
#define __LDBL_DECIMAL_DIG__ 21
#define __LDBL_DENORM_MIN__ 3.64519953188247460253e-4951L
#define __LDBL_DIG__ 18
#define __LDBL_EPSILON__ 1.08420217248550443401e-19L
#define __LDBL_HAS_DENORM__ 1
```

```
#define __LDBL_HAS_INFINITY__ 1
#define __LDBL_HAS_QUIET_NAN__ 1
#define __LDBL_MANT_DIG__ 64
#define __LDBL_MAX_10_EXP__ 4932
#define __LDBL_MAX_EXP__ 16384
#define __LDBL_MAX__ 1.18973149535723176502e+4932L
#define __LDBL_MIN_10_EXP__ (-4931)
#define __LDBL_MIN_EXP__ (-16381)
#define __LDBL_MIN__ 3.36210314311209350626e-4932L
#define __LITTLE_ENDIAN__ 1
#define __LONG_LONG_MAX__ 9223372036854775807LL
#define __LONG_MAX__ 9223372036854775807L
#define __LP64__ 1
#define __MACH__ 1
#define __MMX__ 1
#define __NO_INLINE__ 1
#define __NO_MATH_INLINES 1
#define __OBJC_BOOL_IS_BOOL 0
#define __ORDER_BIG_ENDIAN__ 4321
#define __ORDER_LITTLE_ENDIAN__ 1234
#define __ORDER_PDP_ENDIAN__ 3412
#define __PIC__ 2
#define __POINTER_WIDTH__ 64
#define __PRAGMA_REDEFINE_EXTNAME 1
#define __PTRDIFF_FMTd__ "ld"
#define __PTRDIFF_FMTi__ "li"
#define __PTRDIFF_MAX__ 9223372036854775807L
#define __PTRDIFF_TYPE__ long int
#define __PTRDIFF_WIDTH__ 64
#define __REGISTER_PREFIX__
#define __SCHAR_MAX__ 127
#define __SHRT_MAX__ 32767
#define __SIG_ATOMIC_MAX__ 2147483647
#define __SIG_ATOMIC_WIDTH__ 32
#define __SIZEOF_DOUBLE__ 8
#define __SIZEOF_FLOAT__ 4
#define __SIZEOF_INT128__ 16
#define __SIZEOF_INT__ 4
#define __SIZEOF_LONG_DOUBLE__ 16
#define __SIZEOF_LONG_LONG__ 8
#define __SIZEOF_LONG__ 8
#define __SIZEOF_POINTER__ 8
#define __SIZEOF_PTRDIFF_T__ 8
#define __SIZEOF_SHORT__ 2
```

```
#define __SIZEOF_SIZE_T__ 8
#define __SIZEOF_WCHAR_T__ 4
#define __SIZEOF_WINT_T__ 4
#define __SIZE_FMTX__ "lX"
#define __SIZE_FMTo__ "lo"
#define __SIZE_FMTu__ "lu"
#define __SIZE_FMTx__ "lx"
#define __SIZE_MAX__ 18446744073709551615UL
#define __SIZE_TYPE__ long unsigned int
#define __SIZE_WIDTH__ 64
#define __SSE2_MATH__ 1
#define __SSE2__ 1
#define __SSE3__ 1
#define __SSE_MATH__ 1
#define __SSE__ 1
#define __SSP__ 1
#define __SSSE3__ 1
#define __STDCPP_DEFAULT_NEW_ALIGNMENT__ 16UL
#define __STDC_HOSTED__ 1
#define __STDC_NO_THREADS__ 1
#define __STDC_UTF_16__ 1
#define __STDC_UTF_32__ 1
#define __STDC__ 1
#define __UINT16_C_SUFFIX__
#define __UINT16_FMTX__ "hX"
#define __UINT16_FMTo__ "ho"
#define __UINT16_FMTu__ "hu"
#define __UINT16_FMTx__ "hx"
#define __UINT16_MAX__ 65535
#define __UINT16_TYPE__ unsigned short
#define __UINT32_C_SUFFIX__ U
#define __UINT32_FMTX__ "X"
#define __UINT32_FMTo__ "o"
#define __UINT32_FMTu__ "u"
#define __UINT32_FMTx__ "x"
#define __UINT32_MAX__ 4294967295U
#define __UINT32_TYPE__ unsigned int
#define __UINT64_C_SUFFIX__ ULL
#define __UINT64_FMTX__ "llX"
#define __UINT64_FMTo__ "llo"
#define __UINT64_FMTu__ "llu"
#define __UINT64_FMTx__ "llx"
#define __UINT64_MAX__ 18446744073709551615ULL
#define __UINT64_TYPE__ long long unsigned int
```

```
#define __UINT8_C_SUFFIX__
#define __UINT8_FMTX__ "hhX"
#define __UINT8_FMTo__ "hho"
#define __UINT8_FMTu__ "hhu"
#define __UINT8_FMTx__ "hhx"
#define __UINT8_MAX__ 255
#define __UINT8_TYPE__ unsigned char
#define __UINTMAX_C_SUFFIX__ UL
#define __UINTMAX_FMTX__ "lX"
#define __UINTMAX_FMTo__ "lo"
#define __UINTMAX_FMTu__ "lu"
#define __UINTMAX_FMTx__ "lx"
#define __UINTMAX_MAX__ 18446744073709551615UL
#define __UINTMAX_TYPE__ long unsigned int
#define __UINTMAX_WIDTH__ 64
#define __UINTPTR_FMTX__ "lX"
#define __UINTPTR_FMTo__ "lo"
#define __UINTPTR_FMTu__ "lu"
#define __UINTPTR_FMTx__ "lx"
#define __UINTPTR_MAX__ 18446744073709551615UL
#define __UINTPTR_TYPE__ long unsigned int
#define __UINTPTR_WIDTH__ 64
#define __UINT_FAST16_FMTX__ "hX"
#define __UINT_FAST16_FMTo__ "ho"
#define __UINT_FAST16_FMTu__ "hu"
#define __UINT_FAST16_FMTx__ "hx"
#define __UINT_FAST16_MAX__ 65535
#define __UINT_FAST16_TYPE__ unsigned short
#define __UINT_FAST32_FMTX__ "X"
#define __UINT_FAST32_FMTo__ "o"
#define __UINT_FAST32_FMTu__ "u"
#define __UINT_FAST32_FMTx__ "x"
#define __UINT_FAST32_MAX__ 4294967295U
#define __UINT_FAST32_TYPE__ unsigned int
#define __UINT_FAST64_FMTX__ "lX"
#define __UINT_FAST64_FMTo__ "lo"
#define __UINT_FAST64_FMTu__ "lu"
#define __UINT_FAST64_FMTx__ "lx"
#define __UINT_FAST64_MAX__ 18446744073709551615UL
#define __UINT_FAST64_TYPE__ long unsigned int
#define __UINT_FAST8_FMTX__ "hhX"
#define __UINT_FAST8_FMTo__ "hho"
#define __UINT_FAST8_FMTu__ "hhu"
#define __UINT_FAST8_FMTx__ "hhx"
```

```
#define __UINT_FAST8_MAX__ 255
#define __UINT_FAST8_TYPE__ unsigned char
#define __UINT_LEAST16_FMTX__ "hX"
#define __UINT_LEAST16_FMTo__ "ho"
#define __UINT_LEAST16_FMTu__ "hu"
#define __UINT_LEAST16_FMTx__ "hx"
#define __UINT_LEAST16_MAX__ 65535
#define __UINT_LEAST16_TYPE__ unsigned short
#define __UINT_LEAST32_FMTX__ "X"
#define __UINT_LEAST32_FMTo__ "o"
#define __UINT_LEAST32_FMTu__ "u"
#define __UINT_LEAST32_FMTx__ "x"
#define __UINT_LEAST32_MAX__ 4294967295U
#define __UINT_LEAST32_TYPE__ unsigned int
#define __UINT_LEAST64_FMTX__ "lX"
#define __UINT_LEAST64_FMTo__ "lo"
#define __UINT_LEAST64_FMTu__ "lu"
#define __UINT_LEAST64_FMTx__ "lx"
#define __UINT_LEAST64_MAX__ 18446744073709551615UL
#define __UINT_LEAST64_TYPE__ long unsigned int
#define __UINT_LEAST8_FMTX__ "hhX"
#define __UINT_LEAST8_FMTo__ "hho"
#define __UINT_LEAST8_FMTu__ "hhu"
#define __UINT_LEAST8_FMTx__ "hhx"
#define __UINT_LEAST8_MAX__ 255
#define __UINT_LEAST8_TYPE__ unsigned char
#define __USER_LABEL_PREFIX__ _
#define __VERSION__ "4.2.1 Compatible Apple LLVM 9.0.0 (clang-900.0.39.2)"
#define __WCHAR_MAX__ 2147483647
#define __WCHAR_TYPE__ int
#define __WCHAR_WIDTH__ 32
#define __WINT_TYPE__ int
#define __WINT_WIDTH__ 32
#define __amd64 1
#define __amd64__ 1
#define __apple_build_version__ 9000039
#define __block __attribute__((__blocks__(byref)))
#define __clang__ 1
#define __clang_major__ 9
#define __clang_minor__ 0
#define __clang_patchlevel__ 0
#define __clang_version__ "9.0.0 (clang-900.0.39.2)"
#define __core2 1
#define __core2__ 1
```

```
#define __cplusplus 201103L
#define __cpp_alias_templates 200704
#define __cpp_attributes 200809
#define __cpp_constexpr 200704
#define __cpp_decltype 200707
#define __cpp_delegating_constructors 200604
#define __cpp_exceptions 199711
#define __cpp_inheriting_constructors 201511
#define __cpp_initializer_lists 200806
#define __cpp_lambdas 200907
#define __cpp_nsdmi 200809
#define __cpp_range_based_for 200907
#define __cpp_raw_strings 200710
#define __cpp_ref_qualifiers 200710
#define __cpp_rtti 199711
#define __cpp_rvalue_references 200610
#define __cpp_static_assert 200410
#define __cpp_unicode_characters 200704
#define __cpp_unicode_literals 200710
#define __cpp_user_defined_literals 200809
#define __cpp_variadic_templates 200704
#define __llvm__ 1
#define __nonnull _Nonnull
#define __null_unspecified _Null_unspecified
#define __nullable _Nullable
#define __pic__ 2
#define __private_extern__ extern
#define __strong
#define __tune_core2__ 1
#define __unsafe_unretained
#define __weak __attribute__((objc_gc(weak)))
#define __x86_64 1
#define __x86_64__ 1
```

~~~~~~~~~~~~~~~~~~~~Source code for file moc_restore_wallet.cpp~~~~~~~~~~~~~~~~~~~~

```
/*****************************************************************************
** Meta object code from reading C++ file 'restore_wallet.h'
**
** Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)
**
** WARNING! All changes made in this file will be lost!
*****************************************************************************/
```
~~~~~~~~~~~~~~~~~~~~

```
#include "../Atlas/restore_wallet.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
#if !defined(Q_MOC_OUTPUT_REVISION)
#error "The header file 'restore_wallet.h' doesn't include <QObject>."
#elif Q_MOC_OUTPUT_REVISION != 67
#error "This file was generated using the moc from 5.10.1. It"
#error "cannot be used with the include files from this version of Qt."
#error "(The moc has changed too much.)"
#endif

QT_BEGIN_MOC_NAMESPACE
QT_WARNING_PUSH
QT_WARNING_DISABLE_DEPRECATED
struct qt_meta_stringdata_restore_wallet_t {
    QByteArrayData data[1];
    char stringdata0[15];
};
#define QT_MOC_LITERAL(idx, ofs, len) \
    Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \
    qptrdiff(offsetof(qt_meta_stringdata_restore_wallet_t, stringdata0) + ofs \
        - idx * sizeof(QByteArrayData)) \
    )
static const qt_meta_stringdata_restore_wallet_t qt_meta_stringdata_restore_wallet = {
    {
QT_MOC_LITERAL(0, 0, 14) // "restore_wallet"

    },
    "restore_wallet"
};
#undef QT_MOC_LITERAL

static const uint qt_meta_data_restore_wallet[] = {

 // content:
       7,       // revision
       0,       // classname
       0,    0, // classinfo
       0,    0, // methods
       0,    0, // properties
       0,    0, // enums/sets
       0,    0, // constructors
       0,       // flags
       0,       // signalCount
```

```
    0       // eod
};

void restore_wallet::qt_static_metacall(QObject *_o, QMetaObject::Call _c, int _id, void **_a)
{
   Q_UNUSED(_o);
   Q_UNUSED(_id);
   Q_UNUSED(_c);
   Q_UNUSED(_a);
}

QT_INIT_METAOBJECT const QMetaObject restore_wallet::staticMetaObject = {
   { &QDialog::staticMetaObject, qt_meta_stringdata_restore_wallet.data,
     qt_meta_data_restore_wallet,  qt_static_metacall, nullptr, nullptr}
};


const QMetaObject *restore_wallet::metaObject() const
{
   return QObject::d_ptr->metaObject ? QObject::d_ptr->dynamicMetaObject() :
&staticMetaObject;
}

void *restore_wallet::qt_metacast(const char *_clname)
{
   if (!_clname) return nullptr;
   if (!strcmp(_clname, qt_meta_stringdata_restore_wallet.stringdata0))
      return static_cast<void*>(this);
   return QDialog::qt_metacast(_clname);
}

int restore_wallet::qt_metacall(QMetaObject::Call _c, int _id, void **_a)
{
   _id = QDialog::qt_metacall(_c, _id, _a);
   return _id;
}
QT_WARNING_POP
QT_END_MOC_NAMESPACE
```

~~~~~~~~~~~~~~~~~~~~Source code for file restore_wallet.cpp~~~~~~~~~~~~~~~~~~~~

```
#include "restore_wallet.h"
#include "ui_restore_wallet.h"
```
~~~~~~~~~~~~~~~~~~~~

```
#include <string>

restore_wallet::restore_wallet(QWidget *parent) :
   QDialog(parent),
   ui(new Ui::restore_wallet)
{
   ui->setupUi(this);
}

restore_wallet::~restore_wallet()
{
   delete ui;
}

const std::vector<std::string> restore_wallet::get_word_list()
{
   // Vector holds the word list of the mnemomic 12-word phrase.
   std::vector<std::string> word_list;

   // Append each word to vector.
   word_list.push_back(ui->word_1->text().toStdString());
   word_list.push_back(ui->word_2->text().toStdString());
   word_list.push_back(ui->word_3->text().toStdString());
   word_list.push_back(ui->word_4->text().toStdString());
   word_list.push_back(ui->word_5->text().toStdString());
   word_list.push_back(ui->word_6->text().toStdString());
   word_list.push_back(ui->word_7->text().toStdString());
   word_list.push_back(ui->word_8->text().toStdString());
   word_list.push_back(ui->word_9->text().toStdString());
   word_list.push_back(ui->word_10->text().toStdString());
   word_list.push_back(ui->word_11->text().toStdString());
   word_list.push_back(ui->word_12->text().toStdString());

   return word_list;
}
```

~~~~~~~~~~~~~~~~~~~~~Source code for file app.h~~~~~~~~~~~~~~~~~~~~~

```
#ifndef APP_H
#define APP_H

#include <QErrorMessage>
#include <QMainWindow>
#include "restore_wallet.h"
```
~~~~~~~~~~~~~~~~~~~~~

```
#include "start_menu.h"
#include "../../wallet/stdafx.h"

namespace Ui {
class app;
}

class app : public QMainWindow
{
   Q_OBJECT

public:
   explicit app(QWidget *parent = 0);
   ~app();

private slots:

   // Tab is changed on main application menu.
   void on_tabWidget_tabBarClicked(int index);

   // Copy bitcoin address is clicked.
   void on_copy_btc_address_clicked();

   // Fee slider is moved on send transaction tab.
   void on_fee_slider_sliderMoved(int position);

   // Send transaction button is clicked on send transaction tab.
   void on_send_tx_clicked();

   void on_run_script_btn_clicked();

private:

   // Error Dialog
   QErrorMessage error_msg;

   // Wallet objects.
   Wallet * wallet;
   Network * network;
   Script * script;

   // New or Restore Wallet.
   std::string menu_choice;
```

```
    Ui::app *ui;
    restore_wallet * restore_wallet;
    start_menu * start_menu;

    QLayout * script_layout;

    // Mnemonic word list.
    std::vector<std::string> word_list;

    // Initialize wallet.
    void init_start_menu();
    void init_wallet();
    void get_mnemonic_phrase();

    // Change widgets on main tab.
    void set_main_tab();
    void set_available_payment_address();
    void set_btc_recieved();
    void set_btc_sent();
    void set_btc_balance();

    // Change widgets on send tab.
    void set_send_tab();
    bool send_transaction();

    // Change widgets on history tab.
    void set_history_tab();

    // Input validation for send tab.
    bool is_validate_tx();
    bool is_valid_address();

    // Change widgets on analytics tab.
    void set_analytics_tab();

    // Change widgets on script tab.
    void set_script_tab();
    void run_script();
    void write_to_script_console(std::string msg);

};

#endif // APP_H
```

~~~~~~~~~~~~~~~~~~~~Source code for file start_menu.cpp~~~~~~~~~~~~~~~~~~~~

```
#include "start_menu.h"
#include "ui_start_menu.h"

start_menu::start_menu(QWidget *parent) :
   QDialog(parent),
   ui(new Ui::start_menu)
{
   ui->setupUi(this);
}

start_menu::~start_menu()
{
   delete ui;
}

std::string start_menu::get_menu_choice() const
{
   return menu_choice;
}

void start_menu::on_create_new_wallet_clicked()
{
   menu_choice = NEW_WALLET;
   this->close();
}

void start_menu::on_restore_existing_wallet_clicked()
{
   menu_choice = RESTORE_WALLET;
   this->close();
}
```

~~~~~~~~~~~~~~~~~~~~Source code for file restore_wallet.h~~~~~~~~~~~~~~~~~~~~

```
#ifndef RESTORE_WALLET_H
#define RESTORE_WALLET_H

#include <QDialog>
#include <string>

namespace Ui {
class restore_wallet;
```

```
}

class restore_wallet : public QDialog
{
   Q_OBJECT

public:
   explicit restore_wallet(QWidget *parent = 0);
   ~restore_wallet();

   // Returns a vector of the 12-word phrase.
   const std::vector<std::string> get_word_list();

   // TODO:
   // QValidator, setValidator
   // When value in lineedit is changed, signal is emitted, check if the value change is valid.
   // Require all lines to be filled out.

private:
   Ui::restore_wallet *ui;
};

#endif // RESTORE_WALLET_H
```

~~~~~~~~~~~~~~~~~~~~~Source code for file start_menu.h~~~~~~~~~~~~~~~~~~~~~

```
#ifndef START_MENU_H
#define START_MENU_H

#include <QDialog>

namespace Ui {
class start_menu;
}

class start_menu : public QDialog
{
   Q_OBJECT

public:
   explicit start_menu(QWidget *parent = 0);
   ~start_menu();

   std::string get_menu_choice() const;
```
~~~~~~~~~~~~~~~~~~~~~

```
private slots:

    // New wallet will be made.
    void on_create_new_wallet_clicked();

    // Restore wallet will be made.s
    void on_restore_existing_wallet_clicked();

private:

    const std::string NEW_WALLET = "new";
    const std::string RESTORE_WALLET = "restore";

    std::string menu_choice;

    Ui::start_menu *ui;
};

#endif // START_MENU_H
```

~~~~~~~~~~~~~~~~~~~~Source code for file main.cpp~~~~~~~~~~~~~~~~~~~~

```
#include "app.h"
#include "restore_wallet.h"
#include <QApplication>
#include "start_menu.h"
#include "../../wallet/stdafx.h"

int main(int argc, char *argv[])
{
    // Launch main application window.
    QApplication a(argc, argv);
    app w;
    w.show();

    return a.exec();
}
```

~~~~~~~~~~~~~~~~~~~~Source code for file app.cpp~~~~~~~~~~~~~~~~~~~~

```
#include "app.h"
#include "ui_app.h"
#include "restore_wallet.h"
```

```
#include <string>
#include <vector>
#include <QDebug>
#include "../../wallet/stdafx.h"
#include <QClipboard>
#include <QRadioButton>
#include <qboxlayout.h>

/**
 * @brief Constructor for app::app
 * @param parent
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
app::app(QWidget *parent) :
   QMainWindow(parent),
   ui(new Ui::app)
{
   ui->setupUi(this);

   // Starts immediate dialog box asking user for new/restore wallet.
   init_start_menu();
   init_wallet();
   set_main_tab();

   // Initialize network.
   network = new Network();

   // Initialize script.
   script = new Script();
}

/**
 * @brief Desturctor for app::~app
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
app::~app()
{
   delete ui;
}
```

```
/**
 * @brief Asks user if they would like to start a new wallet or restore wallet. Creates new dialog
window.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::init_start_menu()
{

   // Start new dialog box.
   start_menu = new class start_menu();
   start_menu -> setModal(true);
   start_menu->exec();

   // Get choice from dialog box.
   if(start_menu->close() == true)
   {
      menu_choice = start_menu->get_menu_choice();
   }
   else
   {
      menu_choice = "new";
   }

};

/**
 * @brief Starts new wallet using menmonic seed phrase or restoring an old one from a word
list.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::init_wallet()
{
   // Create new wallet.
   if(menu_choice == "new")
   {
      bc::wallet::word_list mnemonicSeed;
      wallet = new Wallet();
   }
```

```
    // Restore existing wallet.
    else if(menu_choice =="restore")
    {
        get_mnemonic_phrase();
        wallet = new Wallet(word_list);
    }
};

/**
 * @brief Gets mnemonic phrase from the restore wallet dialog box.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::get_mnemonic_phrase()
{
    // Starts new dialog window which asks users for 12-word mnemonic phrase.
    restore_wallet = new class restore_wallet();

    // Main focus of UI will be on new restore_wallet window.
    restore_wallet->setModal(true);

    // Once accepted, get the vector containing phrase.
    if(restore_wallet->exec() == QDialog::Accepted){
        word_list = restore_wallet->get_word_list();
    }
};

/**
 * @brief Sets the text of the availabe payment address.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::set_available_payment_address()
{
    ui->btc_address->setText(QString::fromStdString(wallet->getAddress(1).encoded()));
};

/**
 * @brief Sets the text of the available balance of the bitcoin wallet.
```

```
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::set_btc_balance()
{
   ui->btc_balance->setText(QString::fromStdString(wallet->get_balance_as_string()));
};

/**
 * @brief Refreshes the main tab including the payment addresses and balance.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::set_main_tab()
{
   this->set_available_payment_address();
   this->set_btc_balance();
};

/**
 * @brief Refreshes the analytics tab including the fee recommendations.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::set_analytics_tab()
{
   // Get network fee recommendations.
   network->refreshFeeRecommendations();

   // Change text for fee recommendations.
   ui->fastwait_fee->setText(QString::number(network->getFastestFee()) + " Satoshis per
Byte");
   ui->midwait_fee->setText(QString::number(network->getHalfHourFee()) + " Satoshis per
Byte");
   ui->highwait_fee->setText(QString::number(network->getHourFee()) + " Satoshis per Byte");
};

/**
```

```
 * @brief Refreshes the send transaction tab including the maximum fee that user can send.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::set_send_tab()
{
   // Get network free recommendations.
   network->refreshFeeRecommendations();

   // Set a maximum fee user can send so that the user is not overpaying a fee.
   // This reduces the flexibility of the wallet but places greater important on protecting user
from overpaying fees.
   ui->fee_slider->setMaximum(network->getFastestFee());
};

/**
 * @brief Refreshes the history tab by showing all the last transactions involving wallet's
addresses.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::set_history_tab()
{

   // Vector holding last history of transactions.
   std::vector< std::tuple<unsigned long long,bc::hash_digest,int> > tx =  wallet->
get_transaction_history();

   // Number of transactions.
   const int NUM_TX = tx.size();

   // Select scrollArea.
   QScrollArea *list_of_tx = ui->tx_scroll_area;
   list_of_tx->setBackgroundRole(QPalette::Window);
   list_of_tx->setFrameShadow(QFrame::Plain);
   list_of_tx->setFrameShape(QFrame::NoFrame);
   list_of_tx->setWidgetResizable(true);

   // Array of transactions.
   QGroupBox * transactions[NUM_TX];
```

```
   // Add transaction widget.
   for (int i = 0;i<NUM_TX;i++)
   {
      QGroupBox *groupBox = new QGroupBox(tr("&Transaction"));

      QLabel *date = new QLabel("Block Height: " + QString::number(std::get<2>(tx[i])));
      QLabel *hash = new QLabel("Transaction Hash: " +
QString::fromStdString(bc::encode_hash(std::get<1>(tx[i]))));
      //bc::encode_base10(output.value(), 8)
      QLabel *value = new QLabel("Value: " +
QString::fromStdString(bc::encode_base10(std::get<0>(tx[i]),8)) + " BTC");

      QVBoxLayout *vbox = new QVBoxLayout;
      vbox->addWidget(date);
      vbox->addWidget(hash);
      vbox->addWidget(value);
      vbox->addStretch(1);
      groupBox->setLayout(vbox);
      transactions[i] = groupBox;
   }

   // Add box to main scroll area widget.
   QWidget* boxArea = new QWidget;
   boxArea->setSizePolicy(QSizePolicy::MinimumExpanding, QSizePolicy::MinimumExpanding);
   boxArea->setLayout(new QVBoxLayout(boxArea));
   list_of_tx->setWidget(boxArea);
   QLayout *lay = boxArea->layout();

   // Add transactions to boxArea.
   for (int i = 0;i<NUM_TX;i++)
   {
      lay->addWidget(transactions[i]);
   }
};

/**
 * @brief Refreshes the script tab.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::set_script_tab()
```

```
{
   QScrollArea *script_console = ui->script_scrollArea;

   script_console->setBackgroundRole(QPalette::Window);
   script_console->setFrameShadow(QFrame::Plain);
   script_console->setFrameShape(QFrame::NoFrame);
   script_console->setWidgetResizable(true);

   QWidget* boxArea = new QWidget;
   boxArea->setSizePolicy(QSizePolicy::MinimumExpanding, QSizePolicy::MinimumExpanding);
   boxArea->setLayout(new QVBoxLayout(boxArea));
   script_console->setWidget(boxArea);
   script_layout = boxArea->layout();
};

/**
 * @brief Refreshes a given tab that has been clicked.
 * @param index
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::on_tabWidget_tabBarClicked(int index)
{
   switch(index)
   {
      case 0:
         //main
         this->set_main_tab();
         break;
      case 1:
         //send
         this->set_send_tab();
         break;
      case 2:
         //history
         this->set_history_tab();
         break;
      case 3:
         // fees
         this->set_analytics_tab();
         break;
      case 4:
```

```
        // script
        this->set_script_tab();
        break;
    }
};

/**
 * @brief User clicked on copy bitcoin address. Bitcoin address is copied to computer's
clipboard.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::on_copy_btc_address_clicked()
{
    // Access clipboard.
    QClipboard *clipboard = QApplication::clipboard();
    clipboard->setText(ui->btc_address->text());
};

/**
 * @brief User slided fee slider on send transaction tab.
 * @param position
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::on_fee_slider_sliderMoved(int position)
{
    ui->sat_byte_fee->setText(QString::number(position)+" Satoshis per Byte");
};

/**
 * @brief Performs a check that a given payment address is a legimiate address.
 * @return boolean
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bool
app::is_valid_address()
{
```

```
    std::string address = ui->send_btc_address->toPlainText().toStdString();
    std::cout << address << std::endl;

    /** @todo do appropriate address validation using checksum
     */

    // Bitcoin address is between 26 and 35 characters.
    // check length
    if(address.length()<26 || address.length() >35)
    {
        return false;
    }


    /**
      @todo - check base58 encoding
      */
    else
    {
        return true;
    }
};

/**
 * @brief Sends transaction on send button in send transaction tab. Does validation first.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
app::on_send_tx_clicked()
{
    // validate transaction
    if(this->is_validate_tx())
    {
        //send transaction
        this->send_transaction();
    }

};


/**
 * @brief app::is_validate_tx
```

```
 * @return
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bool
app::is_validate_tx()
{
   //validate btc address
   if(!is_valid_address())
   {

      // Show error message.
      error_msg.showMessage("Invalid address!");

      return false;
   };

   // validate btc amount
   if(wallet->getBalance() < (ui->send_btc_amount->toPlainText().toInt()*100000 + ui-
>fee_slider->value() ))
   {

      std::cout << ui->fee_slider->value() << std::endl;

      // Show error message.
      error_msg.showMessage("Not enough money!");

      return false;
   };

   return true;
};

/**
 * @brief Broadcasts transaction to the network.
 * @return
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bool
app::send_transaction()
{
```

```
    // Get address, value, and fee.
    std::string address = ui->send_btc_address->toPlainText().toStdString();
    unsigned long long amount = ui->send_btc_amount->toPlainText().toInt()*100000;
    unsigned long long fee = ui->fee_slider->value();

    // Broadcast transaction.
    wallet->build_P2PKH(address,amount,fee);
};

/**
 * @brief app::on_run_script_btn_clicked
 *
 * @author Philip Glazman
 * @date 4/30/18
 */
void
app::on_run_script_btn_clicked()
{
    this->run_script();
}

/**
 * @brief app::run_script
 *
 * @author Philip Glazman
 * @date 4/30/18
 */
void
app::run_script()
{
    std::string witness = ui->witness_text_edit->toPlainText().toStdString();
    std::string witness_script = ui->witness_script_text_edit->toPlainText().toStdString();

    script->clear_script();
    script->build_script(witness,witness_script);
    if(script->is_valid())
    {
        this->write_to_script_console("Valid!");
    }
    else
    {
        this->write_to_script_console("Error - Script is invalid.");
    }
};
```

```
/**
 * @brief app::write_to_script_console
 * @param msg
 *
 * @author Philip Glazman
 * @date 5/3/18
 */
void
app::write_to_script_console(std::string msg)
{
   QLabel *console_item = new QLabel(QString::fromStdString(msg));
   script_layout->addWidget(console_item);
};
```

~~~~~~~~~~~~~~~~~~~~Source code for file ui_restore_wallet.h~~~~~~~~~~~~~~~~~~~~

```
/********************************************************************************
***
** Form generated from reading UI file 'restore_wallet.ui'
**
** Created by: Qt User Interface Compiler version 5.10.1
**
** WARNING! All changes made in this file will be lost when recompiling UI file!
********************************************************************************
**/

#ifndef UI_RESTORE_WALLET_H
#define UI_RESTORE_WALLET_H

#include <QtCore/QVariant>
#include <QtWidgets/QAction>
#include <QtWidgets/QApplication>
#include <QtWidgets/QButtonGroup>
#include <QtWidgets/QDialog>
#include <QtWidgets/QDialogButtonBox>
#include <QtWidgets/QHBoxLayout>
#include <QtWidgets/QHeaderView>
#include <QtWidgets/QLabel>
#include <QtWidgets/QLineEdit>
#include <QtWidgets/QVBoxLayout>
#include <QtWidgets/QWidget>

QT_BEGIN_NAMESPACE
```
~~~~~~~~~~~~~~~~~~~~

```
class Ui_restore_wallet
{
public:
    QDialogButtonBox *buttonBox;
    QLabel *label_13;
    QWidget *layoutWidget;
    QHBoxLayout *horizontalLayout_13;
    QVBoxLayout *verticalLayout;
    QHBoxLayout *horizontalLayout;
    QLabel *label;
    QLineEdit *word_1;
    QHBoxLayout *horizontalLayout_2;
    QLabel *label_2;
    QLineEdit *word_2;
    QHBoxLayout *horizontalLayout_3;
    QLabel *label_3;
    QLineEdit *word_3;
    QHBoxLayout *horizontalLayout_4;
    QLabel *label_4;
    QLineEdit *word_4;
    QHBoxLayout *horizontalLayout_5;
    QLabel *label_5;
    QLineEdit *word_5;
    QHBoxLayout *horizontalLayout_6;
    QLabel *label_6;
    QLineEdit *word_6;
    QVBoxLayout *verticalLayout_2;
    QHBoxLayout *horizontalLayout_7;
    QLabel *label_7;
    QLineEdit *word_7;
    QHBoxLayout *horizontalLayout_8;
    QLabel *label_8;
    QLineEdit *word_8;
    QHBoxLayout *horizontalLayout_9;
    QLabel *label_9;
    QLineEdit *word_9;
    QHBoxLayout *horizontalLayout_10;
    QLabel *label_10;
    QLineEdit *word_10;
    QHBoxLayout *horizontalLayout_11;
    QLabel *label_11;
    QLineEdit *word_11;
    QHBoxLayout *horizontalLayout_12;
```

```
QLabel *label_12;
QLineEdit *word_12;

void setupUi(QDialog *restore_wallet)
{
   if (restore_wallet->objectName().isEmpty())
      restore_wallet->setObjectName(QStringLiteral("restore_wallet"));
   restore_wallet->resize(400, 300);
   buttonBox = new QDialogButtonBox(restore_wallet);
   buttonBox->setObjectName(QStringLiteral("buttonBox"));
   buttonBox->setGeometry(QRect(30, 250, 341, 32));
   buttonBox->setOrientation(Qt::Horizontal);
   buttonBox->setStandardButtons(QDialogButtonBox::Cancel|QDialogButtonBox::Ok);
   buttonBox->setCenterButtons(false);
   label_13 = new QLabel(restore_wallet);
   label_13->setObjectName(QStringLiteral("label_13"));
   label_13->setGeometry(QRect(130, 10, 141, 16));
   layoutWidget = new QWidget(restore_wallet);
   layoutWidget->setObjectName(QStringLiteral("layoutWidget"));
   layoutWidget->setGeometry(QRect(40, 30, 312, 218));
   layoutWidget->setAutoFillBackground(false);
   layoutWidget->setInputMethodHints(Qt::ImhNone);
   horizontalLayout_13 = new QHBoxLayout(layoutWidget);
   horizontalLayout_13->setObjectName(QStringLiteral("horizontalLayout_13"));
   horizontalLayout_13->setContentsMargins(0, 0, 0, 0);
   verticalLayout = new QVBoxLayout();
   verticalLayout->setObjectName(QStringLiteral("verticalLayout"));
   horizontalLayout = new QHBoxLayout();
   horizontalLayout->setObjectName(QStringLiteral("horizontalLayout"));
   label = new QLabel(layoutWidget);
   label->setObjectName(QStringLiteral("label"));
   label->setAutoFillBackground(false);
   label->setInputMethodHints(Qt::ImhNone);

   horizontalLayout->addWidget(label);

   word_1 = new QLineEdit(layoutWidget);
   word_1->setObjectName(QStringLiteral("word_1"));
   word_1->setAutoFillBackground(false);
   word_1->setInputMethodHints(Qt::ImhNone);
   word_1->setMaxLength(50);
   word_1->setClearButtonEnabled(false);

   horizontalLayout->addWidget(word_1);
```

```
verticalLayout->addLayout(horizontalLayout);

horizontalLayout_2 = new QHBoxLayout();
horizontalLayout_2->setObjectName(QStringLiteral("horizontalLayout_2"));
label_2 = new QLabel(layoutWidget);
label_2->setObjectName(QStringLiteral("label_2"));
label_2->setAutoFillBackground(false);
label_2->setInputMethodHints(Qt::ImhNone);

horizontalLayout_2->addWidget(label_2);

word_2 = new QLineEdit(layoutWidget);
word_2->setObjectName(QStringLiteral("word_2"));
word_2->setAutoFillBackground(false);
word_2->setInputMethodHints(Qt::ImhNone);
word_2->setMaxLength(50);
word_2->setClearButtonEnabled(false);

horizontalLayout_2->addWidget(word_2);


verticalLayout->addLayout(horizontalLayout_2);

horizontalLayout_3 = new QHBoxLayout();
horizontalLayout_3->setObjectName(QStringLiteral("horizontalLayout_3"));
label_3 = new QLabel(layoutWidget);
label_3->setObjectName(QStringLiteral("label_3"));
label_3->setAutoFillBackground(false);
label_3->setInputMethodHints(Qt::ImhNone);

horizontalLayout_3->addWidget(label_3);

word_3 = new QLineEdit(layoutWidget);
word_3->setObjectName(QStringLiteral("word_3"));
word_3->setAutoFillBackground(false);
word_3->setInputMethodHints(Qt::ImhNone);
word_3->setMaxLength(50);
word_3->setClearButtonEnabled(false);

horizontalLayout_3->addWidget(word_3);
```

```
verticalLayout->addLayout(horizontalLayout_3);

horizontalLayout_4 = new QHBoxLayout();
horizontalLayout_4->setObjectName(QStringLiteral("horizontalLayout_4"));
label_4 = new QLabel(layoutWidget);
label_4->setObjectName(QStringLiteral("label_4"));
label_4->setAutoFillBackground(false);
label_4->setInputMethodHints(Qt::ImhNone);

horizontalLayout_4->addWidget(label_4);

word_4 = new QLineEdit(layoutWidget);
word_4->setObjectName(QStringLiteral("word_4"));
word_4->setAutoFillBackground(false);
word_4->setInputMethodHints(Qt::ImhNone);
word_4->setMaxLength(50);
word_4->setClearButtonEnabled(false);

horizontalLayout_4->addWidget(word_4);


verticalLayout->addLayout(horizontalLayout_4);

horizontalLayout_5 = new QHBoxLayout();
horizontalLayout_5->setObjectName(QStringLiteral("horizontalLayout_5"));
label_5 = new QLabel(layoutWidget);
label_5->setObjectName(QStringLiteral("label_5"));
label_5->setAutoFillBackground(false);
label_5->setInputMethodHints(Qt::ImhNone);

horizontalLayout_5->addWidget(label_5);

word_5 = new QLineEdit(layoutWidget);
word_5->setObjectName(QStringLiteral("word_5"));
word_5->setAutoFillBackground(false);
word_5->setInputMethodHints(Qt::ImhNone);
word_5->setMaxLength(50);
word_5->setClearButtonEnabled(false);

horizontalLayout_5->addWidget(word_5);


verticalLayout->addLayout(horizontalLayout_5);
```

```
horizontalLayout_6 = new QHBoxLayout();
horizontalLayout_6->setObjectName(QStringLiteral("horizontalLayout_6"));
label_6 = new QLabel(layoutWidget);
label_6->setObjectName(QStringLiteral("label_6"));
label_6->setAutoFillBackground(false);
label_6->setInputMethodHints(Qt::ImhNone);

horizontalLayout_6->addWidget(label_6);

word_6 = new QLineEdit(layoutWidget);
word_6->setObjectName(QStringLiteral("word_6"));
word_6->setAutoFillBackground(false);
word_6->setInputMethodHints(Qt::ImhNone);
word_6->setMaxLength(50);
word_6->setClearButtonEnabled(false);

horizontalLayout_6->addWidget(word_6);


verticalLayout->addLayout(horizontalLayout_6);


horizontalLayout_13->addLayout(verticalLayout);

verticalLayout_2 = new QVBoxLayout();
verticalLayout_2->setObjectName(QStringLiteral("verticalLayout_2"));
horizontalLayout_7 = new QHBoxLayout();
horizontalLayout_7->setObjectName(QStringLiteral("horizontalLayout_7"));
label_7 = new QLabel(layoutWidget);
label_7->setObjectName(QStringLiteral("label_7"));
label_7->setAutoFillBackground(false);
label_7->setInputMethodHints(Qt::ImhNone);

horizontalLayout_7->addWidget(label_7);

word_7 = new QLineEdit(layoutWidget);
word_7->setObjectName(QStringLiteral("word_7"));
word_7->setAutoFillBackground(false);
word_7->setInputMethodHints(Qt::ImhNone);
word_7->setMaxLength(50);
word_7->setClearButtonEnabled(false);

horizontalLayout_7->addWidget(word_7);
```

```
verticalLayout_2->addLayout(horizontalLayout_7);

horizontalLayout_8 = new QHBoxLayout();
horizontalLayout_8->setObjectName(QStringLiteral("horizontalLayout_8"));
label_8 = new QLabel(layoutWidget);
label_8->setObjectName(QStringLiteral("label_8"));
label_8->setAutoFillBackground(false);
label_8->setInputMethodHints(Qt::ImhNone);

horizontalLayout_8->addWidget(label_8);

word_8 = new QLineEdit(layoutWidget);
word_8->setObjectName(QStringLiteral("word_8"));
word_8->setAutoFillBackground(false);
word_8->setInputMethodHints(Qt::ImhNone);
word_8->setMaxLength(50);
word_8->setClearButtonEnabled(false);

horizontalLayout_8->addWidget(word_8);


verticalLayout_2->addLayout(horizontalLayout_8);

horizontalLayout_9 = new QHBoxLayout();
horizontalLayout_9->setObjectName(QStringLiteral("horizontalLayout_9"));
label_9 = new QLabel(layoutWidget);
label_9->setObjectName(QStringLiteral("label_9"));
label_9->setAutoFillBackground(false);
label_9->setInputMethodHints(Qt::ImhNone);

horizontalLayout_9->addWidget(label_9);

word_9 = new QLineEdit(layoutWidget);
word_9->setObjectName(QStringLiteral("word_9"));
word_9->setAutoFillBackground(false);
word_9->setInputMethodHints(Qt::ImhNone);
word_9->setMaxLength(50);
word_9->setClearButtonEnabled(false);

horizontalLayout_9->addWidget(word_9);


verticalLayout_2->addLayout(horizontalLayout_9);
```

```
horizontalLayout_10 = new QHBoxLayout();
horizontalLayout_10->setObjectName(QStringLiteral("horizontalLayout_10"));
label_10 = new QLabel(layoutWidget);
label_10->setObjectName(QStringLiteral("label_10"));
label_10->setAutoFillBackground(false);
label_10->setInputMethodHints(Qt::ImhNone);

horizontalLayout_10->addWidget(label_10);

word_10 = new QLineEdit(layoutWidget);
word_10->setObjectName(QStringLiteral("word_10"));
word_10->setAutoFillBackground(false);
word_10->setInputMethodHints(Qt::ImhNone);
word_10->setMaxLength(50);
word_10->setClearButtonEnabled(false);

horizontalLayout_10->addWidget(word_10);


verticalLayout_2->addLayout(horizontalLayout_10);

horizontalLayout_11 = new QHBoxLayout();
horizontalLayout_11->setObjectName(QStringLiteral("horizontalLayout_11"));
label_11 = new QLabel(layoutWidget);
label_11->setObjectName(QStringLiteral("label_11"));
label_11->setAutoFillBackground(false);
label_11->setInputMethodHints(Qt::ImhNone);

horizontalLayout_11->addWidget(label_11);

word_11 = new QLineEdit(layoutWidget);
word_11->setObjectName(QStringLiteral("word_11"));
word_11->setAutoFillBackground(false);
word_11->setInputMethodHints(Qt::ImhNone);
word_11->setMaxLength(50);
word_11->setClearButtonEnabled(false);

horizontalLayout_11->addWidget(word_11);


verticalLayout_2->addLayout(horizontalLayout_11);

horizontalLayout_12 = new QHBoxLayout();
```

```
    horizontalLayout_12->setObjectName(QStringLiteral("horizontalLayout_12"));
    label_12 = new QLabel(layoutWidget);
    label_12->setObjectName(QStringLiteral("label_12"));
    label_12->setAutoFillBackground(false);
    label_12->setInputMethodHints(Qt::ImhNone);

    horizontalLayout_12->addWidget(label_12);

    word_12 = new QLineEdit(layoutWidget);
    word_12->setObjectName(QStringLiteral("word_12"));
    word_12->setAutoFillBackground(false);
    word_12->setInputMethodHints(Qt::ImhNone);
    word_12->setMaxLength(50);
    word_12->setClearButtonEnabled(false);

    horizontalLayout_12->addWidget(word_12);


    verticalLayout_2->addLayout(horizontalLayout_12);


    horizontalLayout_13->addLayout(verticalLayout_2);


    retranslateUi(restore_wallet);
    QObject::connect(buttonBox, SIGNAL(accepted()), restore_wallet, SLOT(accept()));
    QObject::connect(buttonBox, SIGNAL(rejected()), restore_wallet, SLOT(reject()));

    QMetaObject::connectSlotsByName(restore_wallet);
  } // setupUi

  void retranslateUi(QDialog *restore_wallet)
  {
    restore_wallet->setWindowTitle(QApplication::translate("restore_wallet", "Atlas",
nullptr));
    label_13->setText(QApplication::translate("restore_wallet", "Enter 12-word phrase:",
nullptr));
    label->setText(QApplication::translate("restore_wallet", "1.", nullptr));
    word_1->setPlaceholderText(QString());
    label_2->setText(QApplication::translate("restore_wallet", "2.", nullptr));
    word_2->setPlaceholderText(QString());
    label_3->setText(QApplication::translate("restore_wallet", "3.", nullptr));
    word_3->setPlaceholderText(QString());
    label_4->setText(QApplication::translate("restore_wallet", "4.", nullptr));
```

```
        word_4->setPlaceholderText(QString());
        label_5->setText(QApplication::translate("restore_wallet", "5.", nullptr));
        word_5->setPlaceholderText(QString());
        label_6->setText(QApplication::translate("restore_wallet", "6.", nullptr));
        word_6->setPlaceholderText(QString());
        label_7->setText(QApplication::translate("restore_wallet", "7.", nullptr));
        word_7->setPlaceholderText(QString());
        label_8->setText(QApplication::translate("restore_wallet", "8.", nullptr));
        word_8->setPlaceholderText(QString());
        label_9->setText(QApplication::translate("restore_wallet", "9.", nullptr));
        word_9->setPlaceholderText(QString());
        label_10->setText(QApplication::translate("restore_wallet", "10.", nullptr));
        word_10->setPlaceholderText(QString());
        label_11->setText(QApplication::translate("restore_wallet", "11.", nullptr));
        word_11->setPlaceholderText(QString());
        label_12->setText(QApplication::translate("restore_wallet", "12.", nullptr));
        word_12->setPlaceholderText(QString());
    } // retranslateUi

};

namespace Ui {
    class restore_wallet: public Ui_restore_wallet {};
} // namespace Ui

QT_END_NAMESPACE

#endif // UI_RESTORE_WALLET_H
```

~~~~~~~~~~~~~~~~~~~~Source code for file moc_app.cpp~~~~~~~~~~~~~~~~~~~~

```
/*****************************************************************************
** Meta object code from reading C++ file 'app.h'
**
** Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)
**
** WARNING! All changes made in this file will be lost!
*****************************************************************************/

#include "../Atlas/app.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
#if !defined(Q_MOC_OUTPUT_REVISION)
#error "The header file 'app.h' doesn't include <QObject>."
```
~~~~~~~~~~~~~~~~~~~~

```
#elif Q_MOC_OUTPUT_REVISION != 67
#error "This file was generated using the moc from 5.10.1. It"
#error "cannot be used with the include files from this version of Qt."
#error "(The moc has changed too much.)"
#endif

QT_BEGIN_MOC_NAMESPACE
QT_WARNING_PUSH
QT_WARNING_DISABLE_DEPRECATED
struct qt_meta_stringdata_app_t {
    QByteArrayData data[9];
    char stringdata0[146];
};
#define QT_MOC_LITERAL(idx, ofs, len) \
    Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \
    qptrdiff(offsetof(qt_meta_stringdata_app_t, stringdata0) + ofs \
        - idx * sizeof(QByteArrayData)) \
    )
static const qt_meta_stringdata_app_t qt_meta_stringdata_app = {
    {
QT_MOC_LITERAL(0, 0, 3), // "app"
QT_MOC_LITERAL(1, 4, 26), // "on_tabWidget_tabBarClicked"
QT_MOC_LITERAL(2, 31, 0), // ""
QT_MOC_LITERAL(3, 32, 5), // "index"
QT_MOC_LITERAL(4, 38, 27), // "on_copy_btc_address_clicked"
QT_MOC_LITERAL(5, 66, 25), // "on_fee_slider_sliderMoved"
QT_MOC_LITERAL(6, 92, 8), // "position"
QT_MOC_LITERAL(7, 101, 18), // "on_send_tx_clicked"
QT_MOC_LITERAL(8, 120, 25) // "on_run_script_btn_clicked"

    },
    "app\0on_tabWidget_tabBarClicked\0\0index\0"
    "on_copy_btc_address_clicked\0"
    "on_fee_slider_sliderMoved\0position\0"
    "on_send_tx_clicked\0on_run_script_btn_clicked"
};
#undef QT_MOC_LITERAL

static const uint qt_meta_data_app[] = {

 // content:
       7,       // revision
       0,       // classname
       0,    0, // classinfo
```

```
      5,   14, // methods
      0,    0, // properties
      0,    0, // enums/sets
      0,    0, // constructors
      0,       // flags
      0,       // signalCount

 // slots: name, argc, parameters, tag, flags
      1,    1,   39,    2, 0x08 /* Private */,
      4,    0,   42,    2, 0x08 /* Private */,
      5,    1,   43,    2, 0x08 /* Private */,
      7,    0,   46,    2, 0x08 /* Private */,
      8,    0,   47,    2, 0x08 /* Private */,

 // slots: parameters
    QMetaType::Void, QMetaType::Int,    3,
    QMetaType::Void,
    QMetaType::Void, QMetaType::Int,    6,
    QMetaType::Void,
    QMetaType::Void,

      0       // eod
};

void app::qt_static_metacall(QObject *_o, QMetaObject::Call _c, int _id, void **_a)
{
    if (_c == QMetaObject::InvokeMetaMethod) {
        app *_t = static_cast<app *>(_o);
        Q_UNUSED(_t)
        switch (_id) {
        case 0: _t->on_tabWidget_tabBarClicked((*reinterpret_cast< int(*)>(_a[1]))); break;
        case 1: _t->on_copy_btc_address_clicked(); break;
        case 2: _t->on_fee_slider_sliderMoved((*reinterpret_cast< int(*)>(_a[1]))); break;
        case 3: _t->on_send_tx_clicked(); break;
        case 4: _t->on_run_script_btn_clicked(); break;
        default: ;
        }
    }
}

QT_INIT_METAOBJECT const QMetaObject app::staticMetaObject = {
    { &QMainWindow::staticMetaObject, qt_meta_stringdata_app.data,
      qt_meta_data_app,  qt_static_metacall, nullptr, nullptr}
};
```

```
const QMetaObject *app::metaObject() const
{
   return QObject::d_ptr->metaObject ? QObject::d_ptr->dynamicMetaObject() :
&staticMetaObject;
}

void *app::qt_metacast(const char *_clname)
{
   if (!_clname) return nullptr;
   if (!strcmp(_clname, qt_meta_stringdata_app.stringdata0))
      return static_cast<void*>(this);
   return QMainWindow::qt_metacast(_clname);
}

int app::qt_metacall(QMetaObject::Call _c, int _id, void **_a)
{
   _id = QMainWindow::qt_metacall(_c, _id, _a);
   if (_id < 0)
      return _id;
   if (_c == QMetaObject::InvokeMetaMethod) {
      if (_id < 5)
         qt_static_metacall(this, _c, _id, _a);
      _id -= 5;
   } else if (_c == QMetaObject::RegisterMethodArgumentMetaType) {
      if (_id < 5)
         *reinterpret_cast<int*>(_a[0]) = -1;
      _id -= 5;
   }
   return _id;
}
QT_WARNING_POP
QT_END_MOC_NAMESPACE
```

~~~~~~~~~~~~~~~~~~~~Source code for file ui_start_menu.h~~~~~~~~~~~~~~~~~~~~

```
/*********************************************************************
***
** Form generated from reading UI file 'start_menu.ui'
**
** Created by: Qt User Interface Compiler version 5.10.1
**
** WARNING! All changes made in this file will be lost when recompiling UI file!
```
~~~~~~~~~~~~~~~~~~~~

```
*****************************************************************************
**/

#ifndef UI_START_MENU_H
#define UI_START_MENU_H

#include <QtCore/QVariant>
#include <QtWidgets/QAction>
#include <QtWidgets/QApplication>
#include <QtWidgets/QButtonGroup>
#include <QtWidgets/QDialog>
#include <QtWidgets/QHeaderView>
#include <QtWidgets/QPushButton>
#include <QtWidgets/QVBoxLayout>
#include <QtWidgets/QWidget>

QT_BEGIN_NAMESPACE

class Ui_start_menu
{
public:
   QWidget *layoutWidget;
   QVBoxLayout *verticalLayout;
   QPushButton *create_new_wallet;
   QPushButton *restore_existing_wallet;

   void setupUi(QDialog *start_menu)
   {
      if (start_menu->objectName().isEmpty())
         start_menu->setObjectName(QStringLiteral("start_menu"));
      start_menu->resize(400, 300);
      start_menu->setAutoFillBackground(false);
      layoutWidget = new QWidget(start_menu);
      layoutWidget->setObjectName(QStringLiteral("layoutWidget"));
      layoutWidget->setGeometry(QRect(40, 80, 291, 111));
      verticalLayout = new QVBoxLayout(layoutWidget);
      verticalLayout->setObjectName(QStringLiteral("verticalLayout"));
      verticalLayout->setContentsMargins(0, 0, 0, 0);
      create_new_wallet = new QPushButton(layoutWidget);
      create_new_wallet->setObjectName(QStringLiteral("create_new_wallet"));

      verticalLayout->addWidget(create_new_wallet);

      restore_existing_wallet = new QPushButton(layoutWidget);
```

```
        restore_existing_wallet->setObjectName(QStringLiteral("restore_existing_wallet"));

        verticalLayout->addWidget(restore_existing_wallet);


        retranslateUi(start_menu);

        QMetaObject::connectSlotsByName(start_menu);
    } // setupUi

    void retranslateUi(QDialog *start_menu)
    {
        start_menu->setWindowTitle(QApplication::translate("start_menu", "Atlas", nullptr));
        create_new_wallet->setText(QApplication::translate("start_menu", "Create New Wallet",
nullptr));
        restore_existing_wallet->setText(QApplication::translate("start_menu", "Restore Existing
Wallet", nullptr));
    } // retranslateUi

};

namespace Ui {
    class start_menu: public Ui_start_menu {};
} // namespace Ui

QT_END_NAMESPACE

#endif // UI_START_MENU_H
```

~~~~~~~~~~~~~~~~~~~~Source code for file ui_error.h~~~~~~~~~~~~~~~~~~~~

```
/********************************************************************************
***
** Form generated from reading UI file 'error.ui'
**
** Created by: Qt User Interface Compiler version 5.10.1
**
** WARNING! All changes made in this file will be lost when recompiling UI file!
********************************************************************************
**/

#ifndef UI_ERROR_H
#define UI_ERROR_H
```
~~~~~~~~~~~~~~~~~~~~

```
#include <QtCore/QVariant>
#include <QtWidgets/QAction>
#include <QtWidgets/QApplication>
#include <QtWidgets/QButtonGroup>
#include <QtWidgets/QDialog>
#include <QtWidgets/QDialogButtonBox>
#include <QtWidgets/QHeaderView>

QT_BEGIN_NAMESPACE

class Ui_Dialog
{
public:
   QDialogButtonBox *buttonBox;

   void setupUi(QDialog *Dialog)
   {
      if (Dialog->objectName().isEmpty())
         Dialog->setObjectName(QStringLiteral("Dialog"));
      Dialog->resize(400, 300);
      buttonBox = new QDialogButtonBox(Dialog);
      buttonBox->setObjectName(QStringLiteral("buttonBox"));
      buttonBox->setGeometry(QRect(30, 240, 341, 32));
      buttonBox->setOrientation(Qt::Horizontal);
      buttonBox->setStandardButtons(QDialogButtonBox::Cancel|QDialogButtonBox::Ok);

      retranslateUi(Dialog);
      QObject::connect(buttonBox, SIGNAL(accepted()), Dialog, SLOT(accept()));
      QObject::connect(buttonBox, SIGNAL(rejected()), Dialog, SLOT(reject()));

      QMetaObject::connectSlotsByName(Dialog);
   } // setupUi

   void retranslateUi(QDialog *Dialog)
   {
      Dialog->setWindowTitle(QApplication::translate("Dialog", "Dialog", nullptr));
   } // retranslateUi

};

namespace Ui {
   class Dialog: public Ui_Dialog {};
} // namespace Ui
```

```
QT_END_NAMESPACE

#endif // UI_ERROR_H
```

~~~~~~~~~~~~~~~~~~~~Source code for file moc_start_menu.cpp~~~~~~~~~~~~~~~~~~~~

```
/*****************************************************************************
** Meta object code from reading C++ file 'start_menu.h'
**
** Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)
**
** WARNING! All changes made in this file will be lost!
*****************************************************************************/

#include "../Atlas/start_menu.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
#if !defined(Q_MOC_OUTPUT_REVISION)
#error "The header file 'start_menu.h' doesn't include <QObject>."
#elif Q_MOC_OUTPUT_REVISION != 67
#error "This file was generated using the moc from 5.10.1. It"
#error "cannot be used with the include files from this version of Qt."
#error "(The moc has changed too much.)"
#endif

QT_BEGIN_MOC_NAMESPACE
QT_WARNING_PUSH
QT_WARNING_DISABLE_DEPRECATED
struct qt_meta_stringdata_start_menu_t {
   QByteArrayData data[4];
   char stringdata0[76];
};
#define QT_MOC_LITERAL(idx, ofs, len) \
   Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \
   qptrdiff(offsetof(qt_meta_stringdata_start_menu_t, stringdata0) + ofs \
      - idx * sizeof(QByteArrayData)) \
   )
static const qt_meta_stringdata_start_menu_t qt_meta_stringdata_start_menu = {
   {
QT_MOC_LITERAL(0, 0, 10), // "start_menu"
QT_MOC_LITERAL(1, 11, 28), // "on_create_new_wallet_clicked"
QT_MOC_LITERAL(2, 40, 0), // ""
QT_MOC_LITERAL(3, 41, 34) // "on_restore_existing_wallet_cl..."
```
~~~~~~~~~~~~~~~~~~~~

```
    },
    "start_menu\0on_create_new_wallet_clicked\0"
    "\0on_restore_existing_wallet_clicked"
};
#undef QT_MOC_LITERAL

static const uint qt_meta_data_start_menu[] = {

 // content:
       7,       // revision
       0,       // classname
       0,    0, // classinfo
       2,   14, // methods
       0,    0, // properties
       0,    0, // enums/sets
       0,    0, // constructors
       0,       // flags
       0,       // signalCount

 // slots: name, argc, parameters, tag, flags
       1,    0,   24,    2, 0x08 /* Private */,
       3,    0,   25,    2, 0x08 /* Private */,

 // slots: parameters
    QMetaType::Void,
    QMetaType::Void,

       0        // eod
};

void start_menu::qt_static_metacall(QObject *_o, QMetaObject::Call _c, int _id, void **_a)
{
    if (_c == QMetaObject::InvokeMetaMethod) {
        start_menu *_t = static_cast<start_menu *>(_o);
        Q_UNUSED(_t)
        switch (_id) {
        case 0: _t->on_create_new_wallet_clicked(); break;
        case 1: _t->on_restore_existing_wallet_clicked(); break;
        default: ;
        }
    }
    Q_UNUSED(_a);
}
```

```
QT_INIT_METAOBJECT const QMetaObject start_menu::staticMetaObject = {
   { &QDialog::staticMetaObject, qt_meta_stringdata_start_menu.data,
     qt_meta_data_start_menu,  qt_static_metacall, nullptr, nullptr}
};


const QMetaObject *start_menu::metaObject() const
{
   return QObject::d_ptr->metaObject ? QObject::d_ptr->dynamicMetaObject() :
&staticMetaObject;
}

void *start_menu::qt_metacast(const char *_clname)
{
   if (!_clname) return nullptr;
   if (!strcmp(_clname, qt_meta_stringdata_start_menu.stringdata0))
      return static_cast<void*>(this);
   return QDialog::qt_metacast(_clname);
}

int start_menu::qt_metacall(QMetaObject::Call _c, int _id, void **_a)
{
   _id = QDialog::qt_metacall(_c, _id, _a);
   if (_id < 0)
      return _id;
   if (_c == QMetaObject::InvokeMetaMethod) {
      if (_id < 2)
         qt_static_metacall(this, _c, _id, _a);
      _id -= 2;
   } else if (_c == QMetaObject::RegisterMethodArgumentMetaType) {
      if (_id < 2)
         *reinterpret_cast<int*>(_a[0]) = -1;
      _id -= 2;
   }
   return _id;
}
QT_WARNING_POP
QT_END_MOC_NAMESPACE


~~~~~~~~~~~~~~~~~~~~Source code for file ui_app.h~~~~~~~~~~~~~~~~~~~~
/*****************************************************************************
***
** Form generated from reading UI file 'app.ui'
```

```
**
** Created by: Qt User Interface Compiler version 5.10.1
**
** WARNING! All changes made in this file will be lost when recompiling UI file!
********************************************************************************
**/

#ifndef UI_APP_H
#define UI_APP_H

#include <QtCore/QVariant>
#include <QtWidgets/QAction>
#include <QtWidgets/QApplication>
#include <QtWidgets/QButtonGroup>
#include <QtWidgets/QGroupBox>
#include <QtWidgets/QHeaderView>
#include <QtWidgets/QLabel>
#include <QtWidgets/QMainWindow>
#include <QtWidgets/QMenu>
#include <QtWidgets/QMenuBar>
#include <QtWidgets/QPushButton>
#include <QtWidgets/QScrollArea>
#include <QtWidgets/QScrollBar>
#include <QtWidgets/QTabWidget>
#include <QtWidgets/QTextEdit>
#include <QtWidgets/QWidget>

QT_BEGIN_NAMESPACE

class Ui_app
{
public:
    QAction *actionExit;
    QWidget *centralWidget;
    QTabWidget *tabWidget;
    QWidget *tab_main;
    QGroupBox *groupBox;
    QLabel *label;
    QLabel *label_2;
    QLabel *label_3;
    QLabel *btc_recieved;
    QLabel *btc_sent;
    QLabel *btc_balance;
    QGroupBox *groupBox_2;
```

```
QLabel *btc_address;
QPushButton *copy_btc_address;
QWidget *tab_send;
QLabel *label_4;
QLabel *label_5;
QLabel *label_6;
QPushButton *send_tx;
QPushButton *pushButton_2;
QTextEdit *send_btc_address;
QTextEdit *send_btc_amount;
QScrollBar *fee_slider;
QLabel *sat_byte_fee;
QLabel *label_7;
QWidget *tab_history;
QScrollArea *tx_scroll_area;
QWidget *scrollAreaWidgetContents;
QWidget *tab_analytics;
QGroupBox *groupBox_3;
QLabel *label_8;
QLabel *label_9;
QLabel *label_10;
QLabel *fastwait_fee;
QLabel *midwait_fee;
QLabel *highwait_fee;
QWidget *tab_script;
QGroupBox *witness_box;
QTextEdit *witness_text_edit;
QGroupBox *witness_script_box;
QTextEdit *witness_script_text_edit;
QPushButton *run_script_btn;
QScrollArea *script_scrollArea;
QWidget *scrollAreaWidgetContents_2;
QLabel *label_11;
QMenuBar *menuBar;
QMenu *menuAtlas;
QMenu *menuHistory;
QMenu *menuAnalytics;
QMenu *menuScript;

void setupUi(QMainWindow *app)
{
   if (app->objectName().isEmpty())
      app->setObjectName(QStringLiteral("app"));
   app->resize(953, 467);
```

```
app->setTabShape(QTabWidget::Rounded);
actionExit = new QAction(app);
actionExit->setObjectName(QStringLiteral("actionExit"));
centralWidget = new QWidget(app);
centralWidget->setObjectName(QStringLiteral("centralWidget"));
tabWidget = new QTabWidget(centralWidget);
tabWidget->setObjectName(QStringLiteral("tabWidget"));
tabWidget->setGeometry(QRect(10, 30, 931, 401));
tabWidget->setTabsClosable(false);
tab_main = new QWidget();
tab_main->setObjectName(QStringLiteral("tab_main"));
groupBox = new QGroupBox(tab_main);
groupBox->setObjectName(QStringLiteral("groupBox"));
groupBox->setGeometry(QRect(260, 20, 401, 191));
groupBox->setAutoFillBackground(false);
label = new QLabel(groupBox);
label->setObjectName(QStringLiteral("label"));
label->setGeometry(QRect(20, 40, 171, 16));
label_2 = new QLabel(groupBox);
label_2->setObjectName(QStringLiteral("label_2"));
label_2->setGeometry(QRect(20, 70, 171, 16));
label_3 = new QLabel(groupBox);
label_3->setObjectName(QStringLiteral("label_3"));
label_3->setGeometry(QRect(20, 100, 171, 16));
btc_recieved = new QLabel(groupBox);
btc_recieved->setObjectName(QStringLiteral("btc_recieved"));
btc_recieved->setGeometry(QRect(210, 40, 171, 20));
btc_sent = new QLabel(groupBox);
btc_sent->setObjectName(QStringLiteral("btc_sent"));
btc_sent->setGeometry(QRect(210, 70, 181, 16));
btc_balance = new QLabel(groupBox);
btc_balance->setObjectName(QStringLiteral("btc_balance"));
btc_balance->setGeometry(QRect(210, 100, 181, 16));
groupBox_2 = new QGroupBox(tab_main);
groupBox_2->setObjectName(QStringLiteral("groupBox_2"));
groupBox_2->setGeometry(QRect(260, 240, 391, 80));
btc_address = new QLabel(groupBox_2);
btc_address->setObjectName(QStringLiteral("btc_address"));
btc_address->setGeometry(QRect(10, 30, 341, 16));
copy_btc_address = new QPushButton(groupBox_2);
copy_btc_address->setObjectName(QStringLiteral("copy_btc_address"));
copy_btc_address->setGeometry(QRect(270, 50, 113, 32));
tabWidget->addTab(tab_main, QString());
tab_send = new QWidget();
```

```
tab_send->setObjectName(QStringLiteral("tab_send"));
label_4 = new QLabel(tab_send);
label_4->setObjectName(QStringLiteral("label_4"));
label_4->setGeometry(QRect(30, 40, 81, 16));
label_5 = new QLabel(tab_send);
label_5->setObjectName(QStringLiteral("label_5"));
label_5->setGeometry(QRect(30, 80, 81, 16));
label_6 = new QLabel(tab_send);
label_6->setObjectName(QStringLiteral("label_6"));
label_6->setGeometry(QRect(30, 120, 81, 16));
send_tx = new QPushButton(tab_send);
send_tx->setObjectName(QStringLiteral("send_tx"));
send_tx->setGeometry(QRect(140, 150, 113, 32));
pushButton_2 = new QPushButton(tab_send);
pushButton_2->setObjectName(QStringLiteral("pushButton_2"));
pushButton_2->setGeometry(QRect(20, 150, 113, 32));
send_btc_address = new QTextEdit(tab_send);
send_btc_address->setObjectName(QStringLiteral("send_btc_address"));
send_btc_address->setGeometry(QRect(120, 40, 481, 21));
send_btc_address->setInputMethodHints(Qt::ImhNone);
send_btc_amount = new QTextEdit(tab_send);
send_btc_amount->setObjectName(QStringLiteral("send_btc_amount"));
send_btc_amount->setGeometry(QRect(120, 80, 101, 21));
fee_slider = new QScrollBar(tab_send);
fee_slider->setObjectName(QStringLiteral("fee_slider"));
fee_slider->setGeometry(QRect(120, 120, 160, 16));
fee_slider->setMaximum(99);
fee_slider->setOrientation(Qt::Horizontal);
sat_byte_fee = new QLabel(tab_send);
sat_byte_fee->setObjectName(QStringLiteral("sat_byte_fee"));
sat_byte_fee->setGeometry(QRect(300, 120, 131, 16));
label_7 = new QLabel(tab_send);
label_7->setObjectName(QStringLiteral("label_7"));
label_7->setGeometry(QRect(230, 80, 81, 16));
tabWidget->addTab(tab_send, QString());
tab_history = new QWidget();
tab_history->setObjectName(QStringLiteral("tab_history"));
tx_scroll_area = new QScrollArea(tab_history);
tx_scroll_area->setObjectName(QStringLiteral("tx_scroll_area"));
tx_scroll_area->setGeometry(QRect(20, 20, 891, 341));
QSizePolicy sizePolicy(QSizePolicy::Fixed, QSizePolicy::Expanding);
sizePolicy.setHorizontalStretch(0);
sizePolicy.setVerticalStretch(100);
sizePolicy.setHeightForWidth(tx_scroll_area->sizePolicy().hasHeightForWidth());
```

```
tx_scroll_area->setSizePolicy(sizePolicy);
tx_scroll_area->setLineWidth(1);
tx_scroll_area->setVerticalScrollBarPolicy(Qt::ScrollBarAlwaysOn);
tx_scroll_area->setHorizontalScrollBarPolicy(Qt::ScrollBarAlwaysOff);
tx_scroll_area->setSizeAdjustPolicy(QAbstractScrollArea::AdjustIgnored);
tx_scroll_area->setWidgetResizable(true);
scrollAreaWidgetContents = new QWidget();
scrollAreaWidgetContents->setObjectName(QStringLiteral("scrollAreaWidgetContents"));
scrollAreaWidgetContents->setGeometry(QRect(0, 0, 873, 339));
tx_scroll_area->setWidget(scrollAreaWidgetContents);
tabWidget->addTab(tab_history, QString());
tab_analytics = new QWidget();
tab_analytics->setObjectName(QStringLiteral("tab_analytics"));
groupBox_3 = new QGroupBox(tab_analytics);
groupBox_3->setObjectName(QStringLiteral("groupBox_3"));
groupBox_3->setGeometry(QRect(480, 60, 361, 241));
label_8 = new QLabel(groupBox_3);
label_8->setObjectName(QStringLiteral("label_8"));
label_8->setGeometry(QRect(20, 40, 161, 16));
label_9 = new QLabel(groupBox_3);
label_9->setObjectName(QStringLiteral("label_9"));
label_9->setGeometry(QRect(20, 80, 161, 16));
label_10 = new QLabel(groupBox_3);
label_10->setObjectName(QStringLiteral("label_10"));
label_10->setGeometry(QRect(20, 120, 161, 16));
fastwait_fee = new QLabel(groupBox_3);
fastwait_fee->setObjectName(QStringLiteral("fastwait_fee"));
fastwait_fee->setGeometry(QRect(180, 40, 161, 16));
midwait_fee = new QLabel(groupBox_3);
midwait_fee->setObjectName(QStringLiteral("midwait_fee"));
midwait_fee->setGeometry(QRect(180, 80, 161, 16));
highwait_fee = new QLabel(groupBox_3);
highwait_fee->setObjectName(QStringLiteral("highwait_fee"));
highwait_fee->setGeometry(QRect(180, 120, 161, 16));
tabWidget->addTab(tab_analytics, QString());
tab_script = new QWidget();
tab_script->setObjectName(QStringLiteral("tab_script"));
witness_box = new QGroupBox(tab_script);
witness_box->setObjectName(QStringLiteral("witness_box"));
witness_box->setGeometry(QRect(60, 10, 821, 101));
witness_text_edit = new QTextEdit(witness_box);
witness_text_edit->setObjectName(QStringLiteral("witness_text_edit"));
witness_text_edit->setGeometry(QRect(10, 30, 801, 61));
witness_script_box = new QGroupBox(tab_script);
```

```
witness_script_box->setObjectName(QStringLiteral("witness_script_box"));
witness_script_box->setGeometry(QRect(60, 120, 821, 101));
witness_script_text_edit = new QTextEdit(witness_script_box);
witness_script_text_edit->setObjectName(QStringLiteral("witness_script_text_edit"));
witness_script_text_edit->setGeometry(QRect(10, 30, 801, 61));
run_script_btn = new QPushButton(tab_script);
run_script_btn->setObjectName(QStringLiteral("run_script_btn"));
run_script_btn->setGeometry(QRect(60, 220, 113, 32));
script_scrollArea = new QScrollArea(tab_script);
script_scrollArea->setObjectName(QStringLiteral("script_scrollArea"));
script_scrollArea->setGeometry(QRect(70, 300, 801, 71));
script_scrollArea->setWidgetResizable(true);
scrollAreaWidgetContents_2 = new QWidget();
scrollAreaWidgetContents_2-
>setObjectName(QStringLiteral("scrollAreaWidgetContents_2"));
scrollAreaWidgetContents_2->setGeometry(QRect(0, 0, 799, 69));
script_scrollArea->setWidget(scrollAreaWidgetContents_2);
label_11 = new QLabel(tab_script);
label_11->setObjectName(QStringLiteral("label_11"));
label_11->setGeometry(QRect(70, 280, 101, 16));
tabWidget->addTab(tab_script, QString());
app->setCentralWidget(centralWidget);
menuBar = new QMenuBar(app);
menuBar->setObjectName(QStringLiteral("menuBar"));
menuBar->setGeometry(QRect(0, 0, 953, 22));
menuBar->setDefaultUp(true);
menuAtlas = new QMenu(menuBar);
menuAtlas->setObjectName(QStringLiteral("menuAtlas"));
menuHistory = new QMenu(menuBar);
menuHistory->setObjectName(QStringLiteral("menuHistory"));
menuHistory->setAcceptDrops(false);
menuAnalytics = new QMenu(menuBar);
menuAnalytics->setObjectName(QStringLiteral("menuAnalytics"));
menuScript = new QMenu(menuBar);
menuScript->setObjectName(QStringLiteral("menuScript"));
app->setMenuBar(menuBar);

menuBar->addAction(menuAtlas->menuAction());
menuBar->addAction(menuHistory->menuAction());
menuBar->addAction(menuAnalytics->menuAction());
menuBar->addAction(menuScript->menuAction());
menuAtlas->addAction(actionExit);
menuAtlas->addSeparator();
```

```
        retranslateUi(app);

        tabWidget->setCurrentIndex(4);


        QMetaObject::connectSlotsByName(app);
    } // setupUi

    void retranslateUi(QMainWindow *app)
    {
        app->setWindowTitle(QApplication::translate("app", "Atlas", nullptr));
        actionExit->setText(QApplication::translate("app", "Exit", nullptr));
        groupBox->setTitle(QApplication::translate("app", "Overview", nullptr));
        label->setText(QApplication::translate("app", "Recieved:", nullptr));
        label_2->setText(QApplication::translate("app", "Sent:", nullptr));
        label_3->setText(QApplication::translate("app", "Balance:", nullptr));
        btc_recieved->setText(QApplication::translate("app", "btc_recieved", nullptr));
        btc_sent->setText(QApplication::translate("app", "btc_sent", nullptr));
        btc_balance->setText(QApplication::translate("app", "btc_balance", nullptr));
        groupBox_2->setTitle(QApplication::translate("app", "Available Payment Address",
nullptr));
        btc_address->setText(QApplication::translate("app", "btc_address", nullptr));
        copy_btc_address->setText(QApplication::translate("app", "Copy", nullptr));
        tabWidget->setTabText(tabWidget->indexOf(tab_main), QApplication::translate("app",
"Main", nullptr));
        label_4->setText(QApplication::translate("app", "Recipient", nullptr));
        label_5->setText(QApplication::translate("app", "Amount", nullptr));
        label_6->setText(QApplication::translate("app", "Fee", nullptr));
        send_tx->setText(QApplication::translate("app", "Send", nullptr));
        pushButton_2->setText(QApplication::translate("app", "Cancel", nullptr));
        sat_byte_fee->setText(QApplication::translate("app", "0 Satoshis per Byte", nullptr));
        label_7->setText(QApplication::translate("app", "mBTC", nullptr));
        tabWidget->setTabText(tabWidget->indexOf(tab_send), QApplication::translate("app",
"Send", nullptr));
        tabWidget->setTabText(tabWidget->indexOf(tab_history), QApplication::translate("app",
"History", nullptr));
        groupBox_3->setTitle(QApplication::translate("app", "Fees", nullptr));
        label_8->setText(QApplication::translate("app", "Fastest Fee:", nullptr));
        label_9->setText(QApplication::translate("app", "30-minute Fee:", nullptr));
        label_10->setText(QApplication::translate("app", "1-hour Fee:", nullptr));
        fastwait_fee->setText(QApplication::translate("app", "fastwait_fee", nullptr));
        midwait_fee->setText(QApplication::translate("app", "midwait_fee", nullptr));
        highwait_fee->setText(QApplication::translate("app", "highwait_fee", nullptr));
```

```
        tabWidget->setTabText(tabWidget->indexOf(tab_analytics), QApplication::translate("app",
"Analytics", nullptr));
        witness_box->setTitle(QApplication::translate("app", "Unlocking Script (Witness)",
nullptr));
        witness_script_box->setTitle(QApplication::translate("app", "Locking Script (Witness
Script)", nullptr));
        run_script_btn->setText(QApplication::translate("app", "Run", nullptr));
        label_11->setText(QApplication::translate("app", "Script Output", nullptr));
        tabWidget->setTabText(tabWidget->indexOf(tab_script), QApplication::translate("app",
"Script", nullptr));
        menuAtlas->setTitle(QApplication::translate("app", "Atlas", nullptr));
        menuHistory->setTitle(QApplication::translate("app", "History", nullptr));
        menuAnalytics->setTitle(QApplication::translate("app", "Analytics", nullptr));
        menuScript->setTitle(QApplication::translate("app", "Script", nullptr));
    } // retranslateUi

};

namespace Ui {
    class app: public Ui_app {};
} // namespace Ui

QT_END_NAMESPACE

#endif // UI_APP_H
```

~~~~~~~~~~~~~~~~~~~~Source code for file moc_predefs.h~~~~~~~~~~~~~~~~~~~~

```
#define OBJC_NEW_PROPERTIES 1
#define _LP64 1
#define __APPLE_CC__ 6000
#define __APPLE__ 1
#define __ATOMIC_ACQUIRE 2
#define __ATOMIC_ACQ_REL 4
#define __ATOMIC_CONSUME 1
#define __ATOMIC_RELAXED 0
#define __ATOMIC_RELEASE 3
#define __ATOMIC_SEQ_CST 5
#define __BIGGEST_ALIGNMENT__ 16
#define __BLOCKS__ 1
#define __BYTE_ORDER__ __ORDER_LITTLE_ENDIAN__
#define __CHAR16_TYPE__ unsigned short
#define __CHAR32_TYPE__ unsigned int
#define __CHAR_BIT__ 8
```
~~~~~~~~~~~~~~~~~~~~

```
#define __CONSTANT_CFSTRINGS__ 1
#define __DBL_DECIMAL_DIG__ 17
#define __DBL_DENORM_MIN__ 4.9406564584124654e-324
#define __DBL_DIG__ 15
#define __DBL_EPSILON__ 2.2204460492503131e-16
#define __DBL_HAS_DENORM__ 1
#define __DBL_HAS_INFINITY__ 1
#define __DBL_HAS_QUIET_NAN__ 1
#define __DBL_MANT_DIG__ 53
#define __DBL_MAX_10_EXP__ 308
#define __DBL_MAX_EXP__ 1024
#define __DBL_MAX__ 1.7976931348623157e+308
#define __DBL_MIN_10_EXP__ (-307)
#define __DBL_MIN_EXP__ (-1021)
#define __DBL_MIN__ 2.2250738585072014e-308
#define __DECIMAL_DIG__ __LDBL_DECIMAL_DIG__
#define __DEPRECATED 1
#define __DYNAMIC__ 1
#define __ENVIRONMENT_MAC_OS_X_VERSION_MIN_REQUIRED__ 101000
#define __EXCEPTIONS 1
#define __FINITE_MATH_ONLY__ 0
#define __FLT_DECIMAL_DIG__ 9
#define __FLT_DENORM_MIN__ 1.40129846e-45F
#define __FLT_DIG__ 6
#define __FLT_EPSILON__ 1.19209290e-7F
#define __FLT_EVAL_METHOD__ 0
#define __FLT_HAS_DENORM__ 1
#define __FLT_HAS_INFINITY__ 1
#define __FLT_HAS_QUIET_NAN__ 1
#define __FLT_MANT_DIG__ 24
#define __FLT_MAX_10_EXP__ 38
#define __FLT_MAX_EXP__ 128
#define __FLT_MAX__ 3.40282347e+38F
#define __FLT_MIN_10_EXP__ (-37)
#define __FLT_MIN_EXP__ (-125)
#define __FLT_MIN__ 1.17549435e-38F
#define __FLT_RADIX__ 2
#define __FXSR__ 1
#define __GCC_ATOMIC_BOOL_LOCK_FREE 2
#define __GCC_ATOMIC_CHAR16_T_LOCK_FREE 2
#define __GCC_ATOMIC_CHAR32_T_LOCK_FREE 2
#define __GCC_ATOMIC_CHAR_LOCK_FREE 2
#define __GCC_ATOMIC_INT_LOCK_FREE 2
#define __GCC_ATOMIC_LLONG_LOCK_FREE 2
```

```
#define __GCC_ATOMIC_LONG_LOCK_FREE 2
#define __GCC_ATOMIC_POINTER_LOCK_FREE 2
#define __GCC_ATOMIC_SHORT_LOCK_FREE 2
#define __GCC_ATOMIC_TEST_AND_SET_TRUEVAL 1
#define __GCC_ATOMIC_WCHAR_T_LOCK_FREE 2
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_1 1
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_16 1
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_2 1
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_4 1
#define __GCC_HAVE_SYNC_COMPARE_AND_SWAP_8 1
#define __GLIBCXX_BITSIZE_INT_N_0 128
#define __GLIBCXX_TYPE_INT_N_0 __int128
#define __GNUC_GNU_INLINE__ 1
#define __GNUC_MINOR__ 2
#define __GNUC_PATCHLEVEL__ 1
#define __GNUC__ 4
#define __GNUG__ 4
#define __GXX_ABI_VERSION 1002
#define __GXX_EXPERIMENTAL_CXX0X__ 1
#define __GXX_RTTI 1
#define __GXX_WEAK__ 1
#define __INT16_C_SUFFIX__
#define __INT16_FMTd__ "hd"
#define __INT16_FMTi__ "hi"
#define __INT16_MAX__ 32767
#define __INT16_TYPE__ short
#define __INT32_C_SUFFIX__
#define __INT32_FMTd__ "d"
#define __INT32_FMTi__ "i"
#define __INT32_MAX__ 2147483647
#define __INT32_TYPE__ int
#define __INT64_C_SUFFIX__ LL
#define __INT64_FMTd__ "lld"
#define __INT64_FMTi__ "lli"
#define __INT64_MAX__ 9223372036854775807LL
#define __INT64_TYPE__ long long int
#define __INT8_C_SUFFIX__
#define __INT8_FMTd__ "hhd"
#define __INT8_FMTi__ "hhi"
#define __INT8_MAX__ 127
#define __INT8_TYPE__ signed char
#define __INTMAX_C_SUFFIX__ L
#define __INTMAX_FMTd__ "ld"
#define __INTMAX_FMTi__ "li"
```

```
#define __INTMAX_MAX__ 9223372036854775807L
#define __INTMAX_TYPE__ long int
#define __INTMAX_WIDTH__ 64
#define __INTPTR_FMTd__ "ld"
#define __INTPTR_FMTi__ "li"
#define __INTPTR_MAX__ 9223372036854775807L
#define __INTPTR_TYPE__ long int
#define __INTPTR_WIDTH__ 64
#define __INT_FAST16_FMTd__ "hd"
#define __INT_FAST16_FMTi__ "hi"
#define __INT_FAST16_MAX__ 32767
#define __INT_FAST16_TYPE__ short
#define __INT_FAST32_FMTd__ "d"
#define __INT_FAST32_FMTi__ "i"
#define __INT_FAST32_MAX__ 2147483647
#define __INT_FAST32_TYPE__ int
#define __INT_FAST64_FMTd__ "ld"
#define __INT_FAST64_FMTi__ "li"
#define __INT_FAST64_MAX__ 9223372036854775807L
#define __INT_FAST64_TYPE__ long int
#define __INT_FAST8_FMTd__ "hhd"
#define __INT_FAST8_FMTi__ "hhi"
#define __INT_FAST8_MAX__ 127
#define __INT_FAST8_TYPE__ signed char
#define __INT_LEAST16_FMTd__ "hd"
#define __INT_LEAST16_FMTi__ "hi"
#define __INT_LEAST16_MAX__ 32767
#define __INT_LEAST16_TYPE__ short
#define __INT_LEAST32_FMTd__ "d"
#define __INT_LEAST32_FMTi__ "i"
#define __INT_LEAST32_MAX__ 2147483647
#define __INT_LEAST32_TYPE__ int
#define __INT_LEAST64_FMTd__ "ld"
#define __INT_LEAST64_FMTi__ "li"
#define __INT_LEAST64_MAX__ 9223372036854775807L
#define __INT_LEAST64_TYPE__ long int
#define __INT_LEAST8_FMTd__ "hhd"
#define __INT_LEAST8_FMTi__ "hhi"
#define __INT_LEAST8_MAX__ 127
#define __INT_LEAST8_TYPE__ signed char
#define __INT_MAX__ 2147483647
#define __LDBL_DECIMAL_DIG__ 21
#define __LDBL_DENORM_MIN__ 3.64519953188247460253e-4951L
#define __LDBL_DIG__ 18
```

```
#define __LDBL_EPSILON__ 1.08420217248550443401e-19L
#define __LDBL_HAS_DENORM__ 1
#define __LDBL_HAS_INFINITY__ 1
#define __LDBL_HAS_QUIET_NAN__ 1
#define __LDBL_MANT_DIG__ 64
#define __LDBL_MAX_10_EXP__ 4932
#define __LDBL_MAX_EXP__ 16384
#define __LDBL_MAX__ 1.18973149535723176502e+4932L
#define __LDBL_MIN_10_EXP__ (-4931)
#define __LDBL_MIN_EXP__ (-16381)
#define __LDBL_MIN__ 3.36210314311209350626e-4932L
#define __LITTLE_ENDIAN__ 1
#define __LONG_LONG_MAX__ 9223372036854775807LL
#define __LONG_MAX__ 9223372036854775807L
#define __LP64__ 1
#define __MACH__ 1
#define __MMX__ 1
#define __NO_MATH_INLINES 1
#define __OBJC_BOOL_IS_BOOL 0
#define __OPTIMIZE__ 1
#define __ORDER_BIG_ENDIAN__ 4321
#define __ORDER_LITTLE_ENDIAN__ 1234
#define __ORDER_PDP_ENDIAN__ 3412
#define __PIC__ 2
#define __POINTER_WIDTH__ 64
#define __PRAGMA_REDEFINE_EXTNAME 1
#define __PTRDIFF_FMTd__ "ld"
#define __PTRDIFF_FMTi__ "li"
#define __PTRDIFF_MAX__ 9223372036854775807L
#define __PTRDIFF_TYPE__ long int
#define __PTRDIFF_WIDTH__ 64
#define __REGISTER_PREFIX__
#define __SCHAR_MAX__ 127
#define __SHRT_MAX__ 32767
#define __SIG_ATOMIC_MAX__ 2147483647
#define __SIG_ATOMIC_WIDTH__ 32
#define __SIZEOF_DOUBLE__ 8
#define __SIZEOF_FLOAT__ 4
#define __SIZEOF_INT128__ 16
#define __SIZEOF_INT__ 4
#define __SIZEOF_LONG_DOUBLE__ 16
#define __SIZEOF_LONG_LONG__ 8
#define __SIZEOF_LONG__ 8
#define __SIZEOF_POINTER__ 8
```

```
#define __SIZEOF_PTRDIFF_T__ 8
#define __SIZEOF_SHORT__ 2
#define __SIZEOF_SIZE_T__ 8
#define __SIZEOF_WCHAR_T__ 4
#define __SIZEOF_WINT_T__ 4
#define __SIZE_FMTX__ "lX"
#define __SIZE_FMTo__ "lo"
#define __SIZE_FMTu__ "lu"
#define __SIZE_FMTx__ "lx"
#define __SIZE_MAX__ 18446744073709551615UL
#define __SIZE_TYPE__ long unsigned int
#define __SIZE_WIDTH__ 64
#define __SSE2_MATH__ 1
#define __SSE2__ 1
#define __SSE3__ 1
#define __SSE_MATH__ 1
#define __SSE__ 1
#define __SSP__ 1
#define __SSSE3__ 1
#define __STDCPP_DEFAULT_NEW_ALIGNMENT__ 16UL
#define __STDC_HOSTED__ 1
#define __STDC_NO_THREADS__ 1
#define __STDC_UTF_16__ 1
#define __STDC_UTF_32__ 1
#define __STDC__ 1
#define __UINT16_C_SUFFIX__
#define __UINT16_FMTX__ "hX"
#define __UINT16_FMTo__ "ho"
#define __UINT16_FMTu__ "hu"
#define __UINT16_FMTx__ "hx"
#define __UINT16_MAX__ 65535
#define __UINT16_TYPE__ unsigned short
#define __UINT32_C_SUFFIX__ U
#define __UINT32_FMTX__ "X"
#define __UINT32_FMTo__ "o"
#define __UINT32_FMTu__ "u"
#define __UINT32_FMTx__ "x"
#define __UINT32_MAX__ 4294967295U
#define __UINT32_TYPE__ unsigned int
#define __UINT64_C_SUFFIX__ ULL
#define __UINT64_FMTX__ "llX"
#define __UINT64_FMTo__ "llo"
#define __UINT64_FMTu__ "llu"
#define __UINT64_FMTx__ "llx"
```

```
#define __UINT64_MAX__ 18446744073709551615ULL
#define __UINT64_TYPE__ long long unsigned int
#define __UINT8_C_SUFFIX__
#define __UINT8_FMTX__ "hhX"
#define __UINT8_FMTo__ "hho"
#define __UINT8_FMTu__ "hhu"
#define __UINT8_FMTx__ "hhx"
#define __UINT8_MAX__ 255
#define __UINT8_TYPE__ unsigned char
#define __UINTMAX_C_SUFFIX__ UL
#define __UINTMAX_FMTX__ "lX"
#define __UINTMAX_FMTo__ "lo"
#define __UINTMAX_FMTu__ "lu"
#define __UINTMAX_FMTx__ "lx"
#define __UINTMAX_MAX__ 18446744073709551615UL
#define __UINTMAX_TYPE__ long unsigned int
#define __UINTMAX_WIDTH__ 64
#define __UINTPTR_FMTX__ "lX"
#define __UINTPTR_FMTo__ "lo"
#define __UINTPTR_FMTu__ "lu"
#define __UINTPTR_FMTx__ "lx"
#define __UINTPTR_MAX__ 18446744073709551615UL
#define __UINTPTR_TYPE__ long unsigned int
#define __UINTPTR_WIDTH__ 64
#define __UINT_FAST16_FMTX__ "hX"
#define __UINT_FAST16_FMTo__ "ho"
#define __UINT_FAST16_FMTu__ "hu"
#define __UINT_FAST16_FMTx__ "hx"
#define __UINT_FAST16_MAX__ 65535
#define __UINT_FAST16_TYPE__ unsigned short
#define __UINT_FAST32_FMTX__ "X"
#define __UINT_FAST32_FMTo__ "o"
#define __UINT_FAST32_FMTu__ "u"
#define __UINT_FAST32_FMTx__ "x"
#define __UINT_FAST32_MAX__ 4294967295U
#define __UINT_FAST32_TYPE__ unsigned int
#define __UINT_FAST64_FMTX__ "lX"
#define __UINT_FAST64_FMTo__ "lo"
#define __UINT_FAST64_FMTu__ "lu"
#define __UINT_FAST64_FMTx__ "lx"
#define __UINT_FAST64_MAX__ 18446744073709551615UL
#define __UINT_FAST64_TYPE__ long unsigned int
#define __UINT_FAST8_FMTX__ "hhX"
#define __UINT_FAST8_FMTo__ "hho"
```

```
#define __UINT_FAST8_FMTu__ "hhu"
#define __UINT_FAST8_FMTx__ "hhx"
#define __UINT_FAST8_MAX__ 255
#define __UINT_FAST8_TYPE__ unsigned char
#define __UINT_LEAST16_FMTX__ "hX"
#define __UINT_LEAST16_FMTo__ "ho"
#define __UINT_LEAST16_FMTu__ "hu"
#define __UINT_LEAST16_FMTx__ "hx"
#define __UINT_LEAST16_MAX__ 65535
#define __UINT_LEAST16_TYPE__ unsigned short
#define __UINT_LEAST32_FMTX__ "X"
#define __UINT_LEAST32_FMTo__ "o"
#define __UINT_LEAST32_FMTu__ "u"
#define __UINT_LEAST32_FMTx__ "x"
#define __UINT_LEAST32_MAX__ 4294967295U
#define __UINT_LEAST32_TYPE__ unsigned int
#define __UINT_LEAST64_FMTX__ "lX"
#define __UINT_LEAST64_FMTo__ "lo"
#define __UINT_LEAST64_FMTu__ "lu"
#define __UINT_LEAST64_FMTx__ "lx"
#define __UINT_LEAST64_MAX__ 18446744073709551615UL
#define __UINT_LEAST64_TYPE__ long unsigned int
#define __UINT_LEAST8_FMTX__ "hhX"
#define __UINT_LEAST8_FMTo__ "hho"
#define __UINT_LEAST8_FMTu__ "hhu"
#define __UINT_LEAST8_FMTx__ "hhx"
#define __UINT_LEAST8_MAX__ 255
#define __UINT_LEAST8_TYPE__ unsigned char
#define __USER_LABEL_PREFIX__ _
#define __VERSION__ "4.2.1 Compatible Apple LLVM 9.0.0 (clang-900.0.39.2)"
#define __WCHAR_MAX__ 2147483647
#define __WCHAR_TYPE__ int
#define __WCHAR_WIDTH__ 32
#define __WINT_TYPE__ int
#define __WINT_WIDTH__ 32
#define __amd64 1
#define __amd64__ 1
#define __apple_build_version__ 9000039
#define __block __attribute__((__blocks__(byref)))
#define __clang__ 1
#define __clang_major__ 9
#define __clang_minor__ 0
#define __clang_patchlevel__ 0
#define __clang_version__ "9.0.0 (clang-900.0.39.2)"
```

```
#define __core2 1
#define __core2__ 1
#define __cplusplus 201103L
#define __cpp_alias_templates 200704
#define __cpp_attributes 200809
#define __cpp_constexpr 200704
#define __cpp_decltype 200707
#define __cpp_delegating_constructors 200604
#define __cpp_exceptions 199711
#define __cpp_inheriting_constructors 201511
#define __cpp_initializer_lists 200806
#define __cpp_lambdas 200907
#define __cpp_nsdmi 200809
#define __cpp_range_based_for 200907
#define __cpp_raw_strings 200710
#define __cpp_ref_qualifiers 200710
#define __cpp_rtti 199711
#define __cpp_rvalue_references 200610
#define __cpp_static_assert 200410
#define __cpp_unicode_characters 200704
#define __cpp_unicode_literals 200710
#define __cpp_user_defined_literals 200809
#define __cpp_variadic_templates 200704
#define __llvm__ 1
#define __nonnull _Nonnull
#define __null_unspecified _Null_unspecified
#define __nullable _Nullable
#define __pic__ 2
#define __private_extern__ extern
#define __strong
#define __tune_core2__ 1
#define __unsafe_unretained
#define __weak __attribute__((objc_gc(weak)))
#define __x86_64 1
#define __x86_64__ 1
```

~~~~~~~~~~~~~~~~~~~~Source code for file moc_restore_wallet.cpp~~~~~~~~~~~~~~~~~~~~

```
/*****************************************************************************
** Meta object code from reading C++ file 'restore_wallet.h'
**
** Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)
**
** WARNING! All changes made in this file will be lost!
```
~~~~~~~~~~~~~~~~~~~~

```
****************************************************************************/

#include "../Atlas/restore_wallet.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
#if !defined(Q_MOC_OUTPUT_REVISION)
#error "The header file 'restore_wallet.h' doesn't include <QObject>."
#elif Q_MOC_OUTPUT_REVISION != 67
#error "This file was generated using the moc from 5.10.1. It"
#error "cannot be used with the include files from this version of Qt."
#error "(The moc has changed too much.)"
#endif

QT_BEGIN_MOC_NAMESPACE
QT_WARNING_PUSH
QT_WARNING_DISABLE_DEPRECATED
struct qt_meta_stringdata_restore_wallet_t {
    QByteArrayData data[1];
    char stringdata0[15];
};
#define QT_MOC_LITERAL(idx, ofs, len) \
    Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \
    qptrdiff(offsetof(qt_meta_stringdata_restore_wallet_t, stringdata0) + ofs \
        - idx * sizeof(QByteArrayData)) \
    )
static const qt_meta_stringdata_restore_wallet_t qt_meta_stringdata_restore_wallet = {
    {
QT_MOC_LITERAL(0, 0, 14) // "restore_wallet"

    },
    "restore_wallet"
};
#undef QT_MOC_LITERAL

static const uint qt_meta_data_restore_wallet[] = {

 // content:
       7,       // revision
       0,       // classname
       0,    0, // classinfo
       0,    0, // methods
       0,    0, // properties
       0,    0, // enums/sets
       0,    0, // constructors
```

```
      0,      // flags
      0,      // signalCount

      0       // eod
};

void restore_wallet::qt_static_metacall(QObject *_o, QMetaObject::Call _c, int _id, void **_a)
{
   Q_UNUSED(_o);
   Q_UNUSED(_id);
   Q_UNUSED(_c);
   Q_UNUSED(_a);
}

QT_INIT_METAOBJECT const QMetaObject restore_wallet::staticMetaObject = {
   { &QDialog::staticMetaObject, qt_meta_stringdata_restore_wallet.data,
     qt_meta_data_restore_wallet,  qt_static_metacall, nullptr, nullptr}
};


const QMetaObject *restore_wallet::metaObject() const
{
   return QObject::d_ptr->metaObject ? QObject::d_ptr->dynamicMetaObject() :
&staticMetaObject;
}

void *restore_wallet::qt_metacast(const char *_clname)
{
   if (!_clname) return nullptr;
   if (!strcmp(_clname, qt_meta_stringdata_restore_wallet.stringdata0))
      return static_cast<void*>(this);
   return QDialog::qt_metacast(_clname);
}

int restore_wallet::qt_metacall(QMetaObject::Call _c, int _id, void **_a)
{
   _id = QDialog::qt_metacall(_c, _id, _a);
   return _id;
}
QT_WARNING_POP
QT_END_MOC_NAMESPACE
```

~~~~~~~~~~~~~~~~~~~~~Source code for file operation.cpp~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~

```
/**
 * @brief Implementation of Operation class.
 *
 * @file operation.cpp
 * @author Philip Glazman
 * @date 5/3/18
 */

#include "../wallet/stdafx.h"

/**
 * @brief Construct a new Operation:: Operation object
 *
 * @author Philip Glazman
 * @date 5/3/18
 */
Operation::Operation()
{
   // Operation Codes are loaded into hash map for efficient lookup.

   // Insert operation codes for stack manipulation.
   m_op_code_map.emplace("OP_DROP",this->OP_DROP);
   m_op_code_map.emplace("OP_DUP",this->OP_DUP);
   m_op_code_map.emplace("OP_DEPTH",this->OP_DEPTH);

   // Insert operation codes for binary arithmetic.
   m_op_code_map.emplace("OP_EQUAL",this->OP_EQUAL);

   // Insert operation codes for arithmetic.
   m_op_code_map.emplace("OP_1ADD",this->OP_1ADD);
   m_op_code_map.emplace("OP_1SUB",this->OP_1SUB);
   m_op_code_map.emplace("OP_NEGATE",this->OP_NEGATE);
   m_op_code_map.emplace("OP_ABS",this->OP_ABS);
   m_op_code_map.emplace("OP_ADD",this->OP_ADD);
   m_op_code_map.emplace("OP_SUB",this->OP_SUB);
   m_op_code_map.emplace("OP_NUMEQUAL",this->OP_NUMEQUAL);
   m_op_code_map.emplace("OP_NUMNOTEQUAL",this->OP_NUMNOTEQUAL);
   m_op_code_map.emplace("OP_LESSTHAN",this->OP_LESSTHAN);
   m_op_code_map.emplace("OP_GREATERTHAN",this->OP_GREATERTHAN);
   m_op_code_map.emplace("OP_LESSTHANOREQUAL",this->OP_LESSTHANOREQUAL);
   m_op_code_map.emplace("OP_GREATERTHANOREQUAL",this-
>OP_GREATERTHANOREQUAL);
   m_op_code_map.emplace("OP_MIN",this->OP_MIN);
   m_op_code_map.emplace("OP_MAX",this->OP_MAX);
```

```
   m_op_code_map.emplace("OP_WITHIN",this->OP_WITHIN);

   // Insert operation codes for cryptography.
   m_op_code_map.emplace("OP_RIPEMD160",this->OP_RIPEMD160);
   m_op_code_map.emplace("OP_SHA1",this->OP_SHA1);
   m_op_code_map.emplace("OP_SHA256",this->OP_SHA256);
   m_op_code_map.emplace("OP_HASH160",this->OP_HASH160);
   m_op_code_map.emplace("OP_HASH256",this->OP_HASH256);

};

/**
 * @brief Operator for pushing value "1" onto the stack.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_1(std::stack<std::string>& a_stack)
{
   a_stack.push("1");
   return a_stack;
};

/**
 * @brief Operator for pushing an emptry array onto the stack.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
OP_0(std::stack<std::string>& a_stack)
{
   a_stack.push({});
   return a_stack;
};

/**
```

```
 * @brief Operator for pushing empty array onto the stack.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
OP_FALSE(std::stack<std::string>& a_stack)
{
   a_stack.push({});
   return a_stack;
};

/**
 * @brief Operator for popping two top items, adding them, and pushing result onto stack.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
Operation::stack
Operation::OP_ADD(Operation::stack  a_stack)
{
   int x = std::stoi(a_stack.top());
   a_stack.pop();
   int y = std::stoi(a_stack.top());
   a_stack.pop();

   a_stack.push(std::to_string(x+y));
   return a_stack;
};

/**
 * @brief Flips the sign of the top item.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
```

```
 */
std::stack<std::string>&
Operation::OP_NEGATE(std::stack<std::string>& a_stack)
{
   int x = std::stoi(a_stack.top());
   x *= -1;

   a_stack.push(std::to_string(x));
   return a_stack;
};

/**
 * @brief Operator for popping two top items, subtracting them, and pushing result onto stack.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_SUB(std::stack<std::string>& a_stack)
{
   int x = std::stoi(a_stack.top());
   a_stack.pop();
   int y = std::stoi(a_stack.top());
   a_stack.pop();

   // Subtract first from second
   a_stack.push(std::to_string(y-x));
   return a_stack;
};

/**
 * @brief Operator that returns true if top two items are equal numbers.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_NUMEQUAL(std::stack<std::string>& a_stack)
```

```
{
   std::string x = a_stack.top();
   a_stack.pop();
   std::string y = a_stack.top();
   a_stack.pop();

   if(x==y)
   {
      a_stack.push("TRUE");
   }
   else
   {
      a_stack.push("FALSE");
   }
   return a_stack;
};

/**
 * @brief Operator that returns true if top two items are not equal numbers.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_NUMNOTEQUAL(std::stack<std::string>& a_stack)
{
   std::string x = a_stack.top();
   a_stack.pop();
   std::string y = a_stack.top();
   a_stack.pop();

   if(x!=y)
   {
      a_stack.push("TRUE");
   }
   else
   {
      a_stack.push("FALSE");
   }
   return a_stack;
};
```

```
/**
 * @brief Operator for pushing 1 if top two items are equal, push 0 if otherwise.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_EQUAL(std::stack<std::string>& a_stack)
{
   std::string x = a_stack.top();
   a_stack.pop();
   std::string y = a_stack.top();
   a_stack.pop();

   if(x==y)
   {
      a_stack.push("1");
   }
   else
   {
      a_stack.push("0");
   }
   return a_stack;
};

/**
 * @brief Operator for incrementing top value of stack.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_1ADD(std::stack<std::string>& a_stack)
{
   int x = std::stoi(a_stack.top());
   x++;
   a_stack.push(std::to_string(x));
```

```
    return a_stack;
}

/**
 * @brief Operator for decrementing top value of stack.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_1SUB(std::stack<std::string>& a_stack)
{
    int x = std::stoi(a_stack.top());
    x--;
    a_stack.push(std::to_string(x));
    return a_stack;
}

/**
 * @brief Operator for changing the sign of the top item to positive.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_ABS(std::stack<std::string>& a_stack)
{
    int x = std::stoi(a_stack.top());
    x = std::abs(x);
    a_stack.push(std::to_string(x));
    return a_stack;
}

/**
 * @brief Operator that returns true if second item is less than top item.
 *
 * @param a_stack
 * @return std::stack<std::string>&
```

```
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_LESSTHAN(std::stack<std::string>& a_stack)
{
   int x = std::stoi(a_stack.top());
   a_stack.pop();

   int y = std::stoi(a_stack.top());
   a_stack.pop();

   if( y < x)
   {
      a_stack.push("TRUE");
   }
   else
   {
      a_stack.push("FALSE");
   }
   return a_stack;
};

/**
 * @brief Operator that returns true if second item is greater than top item.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_GREATERTHAN(std::stack<std::string>& a_stack)
{
   int x = std::stoi(a_stack.top());
   a_stack.pop();

   int y = std::stoi(a_stack.top());
   a_stack.pop();

   if( y > x)
   {
```

```
        a_stack.push("TRUE");
    }
    else
    {
        a_stack.push("FALSE");
    }
    return a_stack;
};

/**
 * @brief Operator that returns true if second item is less than or equal to top item.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_LESSTHANOREQUAL(std::stack<std::string>& a_stack)
{
    int x = std::stoi(a_stack.top());
    a_stack.pop();

    int y = std::stoi(a_stack.top());
    a_stack.pop();

    if( y <= x)
    {
        a_stack.push("TRUE");
    }
    else
    {
        a_stack.push("FALSE");
    }
    return a_stack;
};

/**
 * @brief Operator that returns true if second item is greater than or equal to top item.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
```

```
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_GREATERTHANOREQUAL(std::stack<std::string>& a_stack)
{
   int x = std::stoi(a_stack.top());
   a_stack.pop();

   int y = std::stoi(a_stack.top());
   a_stack.pop();

   if( y >= x)
   {
      a_stack.push("TRUE");
   }
   else
   {
      a_stack.push("FALSE");
   }
   return a_stack;
};

/**
 * @brief Operator that pushes onto the stack the minimum item of the top two items.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_MIN(std::stack<std::string>& a_stack)
{
   int x = std::stoi(a_stack.top());
   a_stack.pop();

   int y = std::stoi(a_stack.top());
   a_stack.pop();

   if( x < y)
   {
      a_stack.push(std::to_string(x));
```

```
    }
    else
    {
        a_stack.push(std::to_string(y));
    }

    return a_stack;
};

/**
 * @brief Operator that pushes onto the stack the maximum item of the top two items.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_MAX(std::stack<std::string>& a_stack)
{
    int x = std::stoi(a_stack.top());
    a_stack.pop();

    int y = std::stoi(a_stack.top());
    a_stack.pop();

    if( x > y)
    {
        a_stack.push(std::to_string(x));
    }
    else
    {
        a_stack.push(std::to_string(y));
    }

    return a_stack;
};

/**
 * @brief
 *
 * @param a_stack
 * @return std::stack<std::string>&
```

```
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_WITHIN(std::stack<std::string>& a_stack)
{
   int x = std::stoi(a_stack.top());
   a_stack.pop();

   int y = std::stoi(a_stack.top());
   a_stack.pop();

   int z = std::stoi(a_stack.top());
   a_stack.pop();

   if( z >= y && z < x)
   {
      a_stack.push("TRUE");
   }
   else
   {
      a_stack.push("FALSE");
   }

   return a_stack;
};

/**
 * @brief Operator for popping the top item off the stack
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_DROP(std::stack<std::string>& a_stack)
{
   if(!a_stack.empty())
   {
      a_stack.pop();
   };
```

```
    return a_stack;
};

/**
 * @brief Duplicates the tope item in the stack.
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_DUP(std::stack<std::string>& a_stack)
{
    a_stack.push(a_stack.top());
    return a_stack;
};

/**
 * @brief Operator for counting the items on the stack and pushing the result onto the stack
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_DEPTH(std::stack<std::string>& a_stack)
{
    std::string count_stack = std::to_string(a_stack.size());
    a_stack.push(count_stack);
    return a_stack;
};

/**
 * @brief Operator for pushing RIPEMD160 hash of the top item onto the stack
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
```

```
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_RIPEMD160(std::stack<std::string>& a_stack)
{
   std::string hashed_string = hash_RIPEMD160(a_stack.top());

   a_stack.pop();
   a_stack.push(hashed_string);
};

/**
 * @brief Operator for pushing SHA1 hash of the top item onto the stack
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_SHA1(std::stack<std::string>& a_stack)
{
   std::string hashed_string = hash_SHA1(a_stack.top());

   a_stack.pop();
   a_stack.push(hashed_string);
};

/**
 * @brief Operator for pushing SHA256 hash of the top item onto the stack
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_SHA256(std::stack<std::string>& a_stack)
{
   std::string hashed_string = hash_SHA256(a_stack.top());

   a_stack.pop();
```

```
    a_stack.push(hashed_string);
};

/**
 * @brief Operator for pushing RIPEMD160(SHA256(n)) hash of the top item onto the stack
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_HASH160(std::stack<std::string>& a_stack)
{
    // SHA256 hash string
    std::string sha256_hashed_string = hash_SHA256(a_stack.top());

    // Double hash with RIPMD160
    std::string hashed_string = hash_RIPEMD160(sha256_hashed_string);

    a_stack.pop();
    a_stack.push(hashed_string);
};

/**
 * @brief Operator for pushing SHA256(SHA256(n)) hash of the top item onto the stack
 *
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::OP_HASH256(std::stack<std::string>& a_stack)
{
    // SHA256 hash string
    std::string sha256_hashed_string = hash_SHA256(a_stack.top());

    // Double SHA256 hash
    std::string hashed_string = hash_SHA256(sha256_hashed_string);

    a_stack.pop();
```

```
   a_stack.push(hashed_string);
};

/**
 * @brief Calls appropriate operation codes and changes stack.
 *
 * @param a_code
 * @param a_stack
 * @return std::stack<std::string>&
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::stack<std::string>&
Operation::call_operation(std::string a_code,std::stack<std::string> &a_stack)
{
   if(a_code.length() == 1)
   {
      try
      {
         // Change to integer.
         int n = std::stoi(a_code);

         // 1- 75
         if(n > 0 && n < 76)
         {
            a_stack.push(std::to_string(n));
            return a_stack;
         };
      }
      catch(std::exception e)
      {
         std::cout << e.what() << std::endl;
      }
   };


   // Iterator for op code hash map.
   std::unordered_map<std::string,func>::iterator iter;

   iter = m_op_code_map.find(a_code);

   if(iter != m_op_code_map.end())
   {
```

```
        std::cout << "Operation Code Found" << std::endl;
        a_stack = (*iter->second)(a_stack);
    }
    else
    {
        std::cout << "Operation Code NOT Found" << std::endl;
    }

    return a_stack;
};

/**
 * @brief Applies SHA256 hash function on a string.
 *
 * @return std::string
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::string
Operation::hash_SHA256(std::string a_string)
{
    // Digest
    unsigned char digest[SHA256_DIGEST_LENGTH];

    char str[a_string.length()];

    strncpy(str,a_string.c_str(),sizeof(a_string));

    SHA256((unsigned char*)&str,strlen(str),(unsigned char*)&digest);

    char mdString[SHA256_DIGEST_LENGTH*2+1];

    for(int i = 0; i < SHA256_DIGEST_LENGTH; i++)
    {
        sprintf(&mdString[i*2], "%02x", (unsigned int)digest[i]);
    };

    return  std::string(mdString);
};

/**
 * @brief Applies RIPEMD160 hash function a string.
 *
```

```
 * @return std::string
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::string
Operation::hash_RIPEMD160(std::string a_string)
{
   unsigned char digest[RIPEMD160_DIGEST_LENGTH];

   char str[a_string.length()];

   strncpy(str,a_string.c_str(),sizeof(a_string));

   RIPEMD160((unsigned char*)&str,strlen(str),(unsigned char*)&digest);

   char mdString[RIPEMD160_DIGEST_LENGTH*2+1];

   for(int i = 0; i < RIPEMD160_DIGEST_LENGTH; i++)
   {
       sprintf(&mdString[i*2], "%02x", (unsigned int)digest[i]);
   }

   return std::string(mdString);
};

/**
 * @brief Applies SHA1 hash function on string.
 *
 * @return std::string
 *
 * @author Philip Glazman
 * @date 5/2/18
 */
std::string
Operation::hash_SHA1(std::string a_string)
{
   unsigned char digest[SHA_DIGEST_LENGTH];

   char str[a_string.length()];

   strncpy(str,a_string.c_str(),sizeof(a_string));

   SHA1((unsigned char*)&str,strlen(str),(unsigned char*)&digest);
```

```
   char mdString[SHA_DIGEST_LENGTH*2+1];

   for(int i = 0; i < SHA_DIGEST_LENGTH; i++)
   {
      sprintf(&mdString[i*2], "%02x", (unsigned int)digest[i]);
   }

   return std::string(mdString);
};
```

~~~~~~~~~~~~~~~~~~~~Source code for file script.cpp~~~~~~~~~~~~~~~~~~~~

```
/**
 * @brief Implementation of the Script class.
 *
 * @file script.cpp
 * @author Philip Glazman
 * @date 5/3/18
 */
#include "../wallet/stdafx.h"

/**
 * @brief Construct a new script::script object
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
Script::Script()
{
   // Configure current consensus rules for the stack to comply with.
   m_fork_rules = bc::machine::rule_fork::all_rules;

   m_operation = new Operation::Operation();
};

/**
 * @brief Destroy the script::script object
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
Script::~Script()
{
```
~~~~~~~~~~~~~~~~~~~~

```
   delete m_operation;
};


/**
 * @brief Evaluates the script on the current execution stack and returns if script is valid.
 *
 * @author Philip Glazman
 * @date 5/3/18
 */
bool
Script::is_valid()
{
   if(m_execution_stack.size()==1 and (m_execution_stack.top()=="1" ||
m_execution_stack.top()=="True"))
   {
      return true;
   }
   else
   {
      return false;
   }
};

/**
 * @brief
 *
 * @param witness
 * @param witness_script
 * @return true
 * @return false
 *
 * @author Philip Glazman
 * @date 4/30/18
 */
bool
Script::build_script(std::string a_witness, std::string a_witness_script)
{
   std::string script = a_witness +" "+ a_witness_script;

   // Use istringstream class to parse witness and witness script.
   std::istringstream execution_item (script);
```

```
   // Operater/Operand at specific point in script.
   std::string execution_pointer;

   while(execution_item)
   {
      execution_pointer.clear();
      execution_item >> execution_pointer;

      if( execution_pointer != "" )
      {
         // Push the witness onto the execution stack.
         std::cout << execution_pointer << std::endl;
         m_operation->call_operation(execution_pointer,m_execution_stack);
      }
   };
};

/**
 * @brief Clears the current execution stack.
 *
 * @return true
 * @return false
 *
 * @author Philip Glazman
 * @date 5/3/18
 */
bool
Script::clear_script()
{
   while(!m_execution_stack.empty())
   {
      m_execution_stack.pop();
   }

   return true;
};
```

~~~~~~~~~~~~~~~~~~~~~Source code for file debug.cpp~~~~~~~~~~~~~~~~~~~~~

```
#include "../wallet/stdafx.h"

int main()
{
   /*std::stack<std::string> my_stack;
```
~~~~~~~~~~~~~~~~~~~~~

```
    Operation operators;
    operators.call_operation("1",my_stack);
    operators.call_operation("2",my_stack);
    operators.call_operation("OP_ADD",my_stack);
    operators.call_operation("3",my_stack);
    operators.call_operation("OP_EQUAL",my_stack);
    while(!my_stack.empty())
    {
        std::cout << my_stack.top() << std::endl;
        my_stack.pop();
    }*/

    Script my_script;

    std::string witness = "1 2";
    std::string witness_script = "OP_ADD 4 OP_EQUAL";
    my_script.build_script(witness, witness_script);
    std::cout << my_script.is_valid() << std::endl;
}
```

~~~~~~~~~~~~~~~~~~~~Source code for file stdafx.h~~~~~~~~~~~~~~~~~~~~

```
// stdafx.h : include file for standard system include files

/*
TO
compile as g++ -c stdafx.h -o stdafx.h.gch
g++ -c stdafx.h -o stdafx.h.gch -std=c++11 -lboostsystem -lbitcoin
*/

#pragma once

// TODO: add additional headers to the program
// Libbitcoin
#include <bitcoin/bitcoin.hpp>
#include <bitcoin/client.hpp>

// STL
#include <string.h>
#include <iostream>
#include <cstdint>
#include <string>
#include <vector>
#include <iomanip>
```
~~~~~~~~~~~~~~~~~~~~

```
#include <random>
#include <unordered_map>
#include <sstream>
#include <stack>

// Utilities
#include <curl/curl.h>
#include <json/json.h>

// Boost Libraries
#include <boost/property_tree/ptree.hpp>
#include <boost/property_tree/json_parser.hpp>
#include <boost/asio.hpp>

// Atlas Headers
#include "error.hpp"
#include "../network/network.hpp"
#include "../script/script.hpp"
#include "../script/operation.hpp"
#include "Wallet.hpp"
#include "utxo.hpp"
#include "transaction.hpp"

// Crypto Libraries
#include <openssl/ripemd.h>
#include <openssl/sha.h>


~~~~~~~~~~~~~~~~~~~~~Source code for file error.cpp~~~~~~~~~~~~~~~~~~~~~
/*
*   Implementation of the Error class.
*/

#include "stdafx.h"

// Initializes error reports.

std::queue<std::string> Error::m_ErrorMsgs;


/**/
/*
Errors::InitErrorReporting()
NAME
```

```
Errors::InitErrorReporting()
SYNOPSIS
void Errors::InitErrorReporting()
DESCRIPTION
This function empties the error queue in order to remove any junk.
RETURNS
Returns nothing
AUTHOR
Philip Glazman
DATE
1/8/2018
*/
/**/
void
Error::InitErrorReporting()
{
	while (!m_ErrorMsgs.empty()) m_ErrorMsgs.pop();
}

/**/
/*
Errors::RecordError(string &a_emsg)
NAME
Errors::RecordError(string &a_emsg)
SYNOPSIS
void Errors::RecordError(string &a_emsg)
a_emsg		--> Error message to push to the queue.
DESCRIPTION
This function pushes a string error message to the queue.
RETURNS
Returns nothing.
AUTHOR
Philip Glazman
DATE
1/8/2018
*/
/**/
void
Error::RecordError(std::string a_emsg)
{
	m_ErrorMsgs.push(a_emsg);
}

/**/
```

```
/*
Errors::DisplayErrors()
NAME
Errors::DisplayErrors()
SYNOPSIS
void Errors::DisplayErrors()
DESCRIPTION
This function outputs any error messages in the queue.
RETURNS
Returns nothing.
AUTHOR
Philip Glazman
DATE
1/8/2018
*/
/**/
void
Error::DisplayErrors()
{
	// While there are any error messages, print them to the screen.
	while (!m_ErrorMsgs.empty())
	{
		std::cout << std::setw(15) << std::right << m_ErrorMsgs.front() << std::endl;
		m_ErrorMsgs.pop();
	}
}
```

~~~~~~~~~~~~~~~~~~~~Source code for file atlas.cpp~~~~~~~~~~~~~~~~~~~~

```
/*
* Main program for Atlas.
g++ -std=c++11 -o atlaswallet atlas.cpp wallet.cpp error.cpp transaction.cpp
../network/network.cpp utxo.cpp $(pkg-config --cflags libbitcoin --libs libbitcoin libbitcoin-client
libcurl jsoncpp)
*/

#include "stdafx.h"

int
main(int argc, char * argv[])
{
   // Load wallet.
   std::vector< std::string > wordList = {"scatter", "found", "issue", "friend", "front", "glare",
"blanket", "mother", "frequent", "acid", "shaft", "loud"};
```
~~~~~~~~~~~~~~~~~~~~

```
    // Wallet object.
    Wallet wallet(wordList);

    // Reveal keys.
    // wallet.showKeys();

    // Transactions object;
    // Transaction transactions;

    // // Check balance.
    // // int addressIndex = 1;
    // // while(true)
    // // {
    // //     if(transactions.calculateBalance(wallet.getAddress(addressIndex)))
    // //     {
    // //         addressIndex++;
    // //     }
    // //     else
    // //     {
    // //         break;
    // //     }
    // // };

    // // wallet.set_address_index_to_last_unused_address();
    // std::cout<<wallet.getBalance()<<std::endl;
    // std::cout<<wallet.get_balance_as_string()<<std::endl;

    // std::cout<<wallet.getAddress(1).is_address()<<std::endl;

    // std::cout << addressIndex << std::endl;
    // std::cout << transactions.getBalance() << std::endl;

    // bc::wallet::payment_address addy = wallet.getAddress(1);
    // bc::wallet::payment_address destinationAddy = wallet.getAddress(3);
    // bc::data_chunk publicKey = bc::to_chunk(wallet.childPublicKey(1).point());
    // wallet.build_P2PKH("mmUbEcLMoJsaT6Uy3ZBkvF5i1AJ5xgmZpG",1000000);
    // transactions.P2PKH(destinationAddy, 1000000);

    // // Fees
    // Network net;
    // net.refreshFeeRecommendations();

};
```

~~~~~~~~~~~~~~~~~~~~~Source code for file utxo.cpp~~~~~~~~~~~~~~~~~~~~~

```
#include "stdafx.h"

/**
 * @brief Construct a new utxo::utxo object
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
utxo::utxo()
{
   m_tx_output = new std::vector < std::tuple <m_satoshis, m_utxo_hash, m_address> >;
};

/**
 * @brief Destroy the utxo::utxo object
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
utxo::~utxo()
{
   delete m_tx_output;
};

/**
 * @brief Adds a transaction to the utxo map.
 *
 * @param a_satoshis
 * @param a_utxo_hash
 * @param a_address
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void utxo::add_transaction(unsigned long long a_satoshis, bc::hash_digest a_utxo_hash,
bc::wallet::payment_address a_address) const
{
   m_tx_output -> push_back( std::make_tuple(a_satoshis, a_utxo_hash, a_address));
}
```
~~~~~~~~~~~~~~~~~~~~~

```
/**
 * @brief Returns value of a transaction hash.
 *
 * @param a_utxo_hash
 * @return unsigned long long
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
unsigned long long utxo::get_value(bc::hash_digest a_utxo_hash)
{
   //@TODO - update
   return 1;
};


// bool compare_utxo(const utxo_tuple& lhs, const utxo_tuple& rhs)
// {
//     return std::get<0>(lhs) <  std::get<0>(rhs);
// };



void utxo::show_available_utxo()
{
   for(const auto&tx : *m_tx_output)
   {
      std::cout << "Payment Address: " << std::get<2>(tx) << " Value: " << std::get<0>(tx) << "
UTXO Hash: " << bc::encode_hash(std::get<1>(tx)) << std::endl;
   };
};

// finds the minimum utxo to satisfy need
// returns stack of utxos which conntains tuple of payment address, utxo hash, and value
utxo::utxo_data utxo::find_utxo(unsigned long long a_satoshis)
{
   // Used to sum each utxo value
   unsigned long long value = 0;
   utxo_data utxo_to_return;

   // Sort vector in ascending order (min to max) according to utxo value.
   std::sort(m_tx_output->begin(), m_tx_output->end(),compare_utxo());
```

```
    show_available_utxo();

    for(const auto&tx : *m_tx_output)
    {
        if(value > a_satoshis)
        {
            break;
        }
        utxo_to_return.push_back(tx);
        value += std::get<0>(tx);
    };

    return utxo_to_return;
};
// get utxo based on what to spend - get lowest
```

~~~~~~~~~~~~~~~~~~~~Source code for file Wallet.cpp~~~~~~~~~~~~~~~~~~~~

```
#include "stdafx.h"


/**
 * @brief Creates new wallet using user entropy (256 bits).
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
Wallet::Wallet()
{
    bc::wallet::word_list mnemonicSeed = generateMnemonicCode();

    m_seed = bc::to_chunk(bc::wallet::decode_mnemonic(mnemonicSeed));

    m_mnemonic = mnemonicSeed;

    // Master 256-bit Private Key.
    m_masterPrivateKey = bc::wallet::hd_private(m_seed,bc::wallet::hd_private::testnet);

    // Master 264-bit Public Key.
    m_masterPublicKey = m_masterPrivateKey.to_public();

    // Transactions object.
    transactions = new Transaction();
    m_address_index=1;
```
~~~~~~~~~~~~~~~~~~~~

```
    set_address_index_to_last_unused_address();
}

/**
 * @brief Creates new wallet by import 12 word phrase.
 *
 * @param a_mnemonicSeed, bc::wallet::word_list. List of 12 word seed phrase.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
Wallet::Wallet(const bc::wallet::word_list a_mnemonicSeed)
{
    // 512 bit seed is derived from mnemonic bits.
    m_seed = bc::to_chunk(bc::wallet::decode_mnemonic(a_mnemonicSeed));

    m_mnemonic = a_mnemonicSeed;

    // Master 256-bit Private Key.
    m_masterPrivateKey = bc::wallet::hd_private(m_seed,bc::wallet::hd_private::testnet);

    // Master 264-bit Public Key.
    m_masterPublicKey = m_masterPrivateKey.to_public();

    // Transactions object.
    transactions = new Transaction();
    m_address_index=1;
    set_address_index_to_last_unused_address();
}

/**
 * @brief Generates mnemonic bits using user machine's entropy. BIP-39 Standard.
 *
 * @return bc::wallet::word_list. List of 12 words representing seed of wallet.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::wallet::word_list
Wallet::generateMnemonicCode()
{
    // Store 128 bits for entropy.
    m_entropy = new std::vector<std::uint8_t>(16);
```

```
    // Entropy is generated using local machine.
    bc::pseudo_random_fill(*m_entropy);

    // Entropy is included in bits to generate mnemonic words.
    bc::wallet::word_list mnemonicSeed =  bc::wallet::create_mnemonic(*m_entropy);

    delete m_entropy;

    return mnemonicSeed;

    // Create 512-bit seed using mnemonic code wirds and a_passphrase as Salt.
    // TODO - add ICU to library dependency to make it work with passphrase
};

/**
 * @brief Selector for child private key at index n of keychain.
 *
 * @param a_index, integer.
 * @return bc::wallet::hd_private
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::wallet::hd_private
Wallet::childPrivateKey(int a_index)
{
    return m_masterPrivateKey.derive_private(a_index);
}

/**
 * @brief Selector for child public key at index n of keychain.
 *
 * @param a_index, integer.
 * @return bc::wallet::hd_public
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::wallet::hd_public
Wallet::childPublicKey(int a_index)
{
    return m_masterPublicKey.derive_public(a_index);
}
```

```
/**
 * @brief Return the Bitcoin Address (Base58 encoded address) at index n of keychain.
 *
 * @param a_index, integer.
 * @return bc::wallet::payment_address
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::wallet::payment_address Wallet::childAddress(int a_index)
{
   // Testnet payment address.
   return bc::wallet::payment_address(bc::wallet::ec_public(childPublicKey(a_index).point()),
0x6f);
}

/**
 * @brief Returns BIP-32 root key.
 *
 * @return bc::wallet::hd_private
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::wallet::hd_private Wallet::showPrivateKey()
{
   return m_masterPrivateKey.encoded();
}

/**
 * @brief Returns child private key at index n of keychain.
 *
 * @param index
 * @return bc::wallet::hd_private
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::wallet::hd_private Wallet::showChildPrivateKey(int a_index)
{
   return childPrivateKey(a_index).encoded();
}

/**
```

```
 * @brief Return bitcoin address (Base58 encoded) at index n of keychain.
 *
 * @param a_index
 * @return bc::wallet::payment_address
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::wallet::payment_address
Wallet::getAddress(int a_index)
{
   return childAddress(a_index).encoded();
}

/**
 * @brief Outputs to console the list of mnemonic code phrases.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
Wallet::showMnemonicCodes()
{
   // Validate the mnemonic phrase before sharing it with user.
   if(bc::wallet::validate_mnemonic(m_mnemonic))
   {
      std::string mnemonicString = bc::join(m_mnemonic);
      std::cout << mnemonicString << std::endl;

   }else{
      std::cout << "Mnemonic Invalid!" << std::endl;
   }
};

/**
 * @brief Shows relevant keys to the user in console. Used for debugging.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
Wallet::showKeys()
{
   showMnemonicCodes();
```

```
   std::cout << "BIP 32 Root Key: " << showPrivateKey() << std::endl;
   std::cout << "Address: " << getAddress(1) << std::endl;
   std::cout << "Address: " << getAddress(2) << std::endl;
};

/**
 * @brief Sets the current address index to the last unused address. Prevents address reuse.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
Wallet::set_address_index_to_last_unused_address()
{
   while(true)
   {
      // Check if the given address was used.
      if(transactions->calculateBalance(getAddress(m_address_index)))
      {
         m_address_index++;
      }
      else
      {
         break;
      }
   }
};

/**
 * @brief Returns balance as unsigned long long.
 *
 * @return unsigned long long represents balance value of wallet.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
unsigned long long
Wallet::getBalance() const
{
   return transactions->getBalance();
}

/**
 * @brief Returns balance as string.
```

```
 *
 * @return std::string string that represents balance value of wallet.
 * @author Philip Glazman
 * @date 4/28/18
 */
std::string
Wallet::get_balance_as_string() const
{
   return bc::encode_base10(transactions->getBalance(),8);
};

/**
 * @brief Creates a P2PKH transaction
 *
 * @param a_address string Address to send value to.
 * @param a_satoshis unsigned long long Satoshi value to send.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
Wallet::build_P2PKH(std::string a_address, unsigned long long a_satoshis)
{
   // Build tx.
   bc::wallet::payment_address address = bc::wallet::payment_address(a_address);
   bc::chain::transaction tx = transactions->P2PKH(a_address,a_satoshis);

   // Show tx.
   // @TODO - return tx.
   transactions->show_raw_tx(tx);
   // transactions->broadcastTransaction(tx);
};

/**
 * @brief Creates a P2PKH transaction with a given tx fee.
 *
 * @param a_address string Address to send value to.
 * @param a_satoshis unsigned long long Satoshi value to send.
 * @param a_fees
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
```

```
Wallet::build_P2PKH(std::string a_address, unsigned long long a_satoshis, unsigned long long
a_fees)
{
   // Build tx.
   bc::wallet::payment_address address = bc::wallet::payment_address(a_address);
   bc::chain::transaction tx = transactions->P2PKH(a_address,a_satoshis,a_fees);

   transactions->show_raw_tx(tx);
   std::cout << tx.inputs()[0].address() << std::endl;
   const bc::wallet::payment_address utxo_address = transactions->get_last_utxo_address();

   bc::data_chunk public_key = getPublicKey(utxo_address);
   bc::wallet::hd_private private_key = getPrivateKey(utxo_address);

   bc::endorsement signature = transactions->create_signature(public_key,private_key,tx);
   bc::chain::script unlocking_script = transactions->create_sig_script(signature,public_key);
   tx.inputs()[0].set_script(unlocking_script);


   transactions->show_raw_tx(tx);


   // Broadcast tx.
   // transactions->broadcastTransaction(tx);
};

/**
 * @brief
 *
 * @return std::vector< Transaction::m_tx >
 *
 * @author Philip Glazman
 * date 4/28/18
 */
std::vector< Transaction::m_tx >
Wallet::get_transaction_history()
{
   return transactions->get_transaction_history();
}

/**
 * @brief Returns public key with a given payment address.
 *
 * @param a_address
```

```
 * @return bc::data_chunk
 *
 * @author Philip Glazman
 * @date 4/29/18
 */
bc::data_chunk
Wallet::getPublicKey(bc::wallet::payment_address a_address)
{

   for(int i = 1 ; i  < INT_MAX; i ++ )
   {
      if(childAddress(i) == a_address)
      {
         return bc::to_chunk(childPublicKey(i).point());
      }
   }

};

/**
 * @brief Returns private key with a given payment address.
 *
 * @param a_address
 * @return bc::wallet::hd_private
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::wallet::hd_private
Wallet::getPrivateKey(bc::wallet::payment_address a_address)
{

   for(int i = 1 ; i  < INT_MAX; i ++ )
   {
      if(childAddress(i) == a_address)
      {
         //childPrivateKey
         return childPrivateKey(i);
      }
   }

};
```

~~~~~~~~~~~~~~~~~~~~~Source code for file transaction.cpp~~~~~~~~~~~~~~~~~~~~~

```
#include "stdafx.h"

/**
 * @brief Construct a new Transaction:: Transaction object
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
Transaction::Transaction()
{
   network = new Network();
   unspent_output = new utxo();

   m_utxoSum = 0;
};

/**
 * @brief Destroy the Transaction:: Transaction object
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
Transaction::~Transaction()
{
   delete network;
   delete unspent_output;
}

/**
 * @brief Creates output for a P2PKH transaction.
 *
 * @param a_address bc::wallet::payment_address address that owns output
 * @param a_satoshis unsigned long long value of output
 * @return bc::chain::output
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::chain::output
Transaction::createOutputP2PKH(bc::wallet::payment_address a_address, unsigned long long
a_satoshis)
{
```
~~~~~~~~~~~~~~~~~~~~~

```
   // Hash the Public Key of the Address. OP_DUP OP_HASH160 <PKH> OP_EQUALVERIFY
OP_CHECKSIG
   bc::chain::script outputScript  =
bc::chain::script().to_pay_key_hash_pattern(a_address.hash());

   // to_pay_key_hash_pattern creates an operation::list. Assignment constructor makes
assigns it to outputScript.
   bc::chain::output output(a_satoshis,outputScript);

   return output;
};

/**
 * @brief Shows the transaction output.
 *
 * @param output bc::chain::output output point
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
Transaction::showTxOutput(bc::chain::output output)
{
   std::cout << "Sending Bitcoin: \nAmount: " << bc::encode_base10(output.value(), 8) << "BTC :
Output Script: " << output.script().to_string(0) << std::endl;
};

/**
 * @brief Outputs raw transaction into hex.
 *
 * @param a_transaction
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
Transaction::show_raw_tx(bc::chain::transaction a_transaction)
{
   std::cout << "Raw Transaction: " << std::endl;
          std::cout << bc::encode_base16(a_transaction.to_data()) << std::endl;
}

/**
```

```
 * @brief Creates an approximate size of the transaction in bytes using the number of inputs
and outputs.
 *
 * @param inputs, integer. Number of inputs in the transaction.
 * @param outputs, integer. Number of outputs in the transaction.
 * @return int, Number of bytes.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
int
Transaction::calculateTxSize(int inputs, int outputs)
{
   // Conservative case, inputs are 181 bytes. Uncompressed public keys vary in size.
   // Outputs are 34 bytes.
   return inputs*181+outputs*34+10;
};

/**
 * @brief Calculates transaction fee for a given transaction size.
 *
 * @param estimated_tx_size int size of the transaction in bytes.
 * @return unsigned long long
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
unsigned long long
Transaction::calculate_tx_fee(int estimated_tx_size)
{
   // Refresh fee recommendations.
   network->refreshFeeRecommendations();

   // Satoshis/Bytes * Bytes
   unsigned long long fees = (unsigned long long)(estimated_tx_size * network->getHourFee());

   return fees;
};

/**
 * @brief Create a Meta Data Tx object
 *
 * @return true
 * @return false
```

```
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bool
createMetaDataTx()
{
   // OP Return tx
   std::string messageString = "helloworld";
   bc::data_chunk data(80);
   auto source = bc::make_safe_deserializer(data.begin(),data.end());
   auto sink = bc::make_unsafe_serializer(data.begin());
   sink.write_string(messageString);
   const auto nullData = source.read_bytes(80);
   std::cout << "Message: " << std::endl;
         std::cout << bc::encode_base16(nullData) << std::endl;
   bc::chain::output output2 = bc::chain::output();
   output2.set_script(bc::chain::script(bc::chain::script().to_null_data_pattern(nullData)));
   output2.set_value(0);
   return true;
};

/**
 * @brief Create a Signature object
 *
 * @param a_pubKey
 * @param a_privKey
 * @param a_transaction
 * @return bc::endorsement
 *
 * @author Philip Glazman
 * @date 4/29/18
 */
bc::endorsement
Transaction::create_signature(bc::data_chunk a_pubKey, bc::ec_secret
a_privKey,bc::chain::transaction a_transaction)
{
   bc::chain::script lockingScript =
bc::chain::script().to_pay_key_hash_pattern(bc::bitcoin_short_hash(a_pubKey));
   bc::endorsement signature;

   if(lockingScript.create_endorsement(signature, a_privKey, lockingScript, a_transaction, 0u,
bc::machine::all))
         {
```

```
                std::cout << "Signature: " << std::endl;
                std::cout << bc::encode_base16(signature) << "\n" << std::endl;
    return signature;
        }
  else
  {
    Error::RecordError(std::string("Cannot create signature endorsement."));
    Error::DisplayErrors();
  }
};


/**
 * @brief Create a sig script object
 *
 * @param a_signature
 * @param a_pubKey
 * @return bc::script
 *
 * @author Philip Glazman
 * @date 4/29/18
 */
bc::chain::script
Transaction::create_sig_script(bc::endorsement a_signature, bc::data_chunk a_pubKey)
{
  bc::machine::operation::list signature_script;
  signature_script.push_back(bc::machine::operation(a_signature));
  signature_script.push_back(bc::machine::operation(a_pubKey));
  bc::chain::script unlocking_script(signature_script);
  return unlocking_script;
}

/**
 * @brief Constructs P2PKH script transaction.
 *
 * @param a_publicKey, public key address of bitcoin payment address to use.
 * @param a_privKey
 * @param a_destinationAddress
 * @param a_satoshis
 * @return true
 * @return false
 *
 * @author Philip Glazman
 * @date 4/28/18
```

```
 */
bc::chain::transaction
Transaction::P2PKH(bc::wallet::payment_address a_destinationAddress, unsigned long long
a_satoshis)
{
   unsigned long long input_value = 0;
   unsigned long long change_value = 0;

   // Start building Transaction.
   // Instantiate the transaction object.
   bc::chain::transaction tx = bc::chain::transaction();

   // Set transaction version.
   // uint32_t version = 1u;
   // tx.set_version(version);

   // Find unspent output.
   m_utxo utxo_to_spend = unspent_output -> find_utxo(a_satoshis);

   // Create inputs.
   // For each input, point to unspent transaction output.
   for( const auto &utxo : utxo_to_spend)
   {
      // Create input.
      bc::chain::input input = bc::chain::input();
      bc::hash_digest utxo_hash = std::get<1>(utxo);
      bc::chain::output_point previous_output(utxo_hash,0);

      input.set_previous_output(previous_output);
      input.set_sequence(0xffffffff);
      tx.inputs().push_back(input);

      input_value += std::get<0>(utxo);

      // Get pub_key.
      m_last_utxo_address = std::get<2>(utxo);
   };

   change_value = input_value - a_satoshis;

   // Find recommended fee.
   int estimated_tx_bytes = calculateTxSize(tx.inputs().size(), 2);
   unsigned long long fees = calculate_tx_fee(estimated_tx_bytes);
```

```
    // Subtract fees from the change.
    if( change_value - fees > 0)
    {
        change_value -= fees;
        std::cout << "fees to send: " <<fees << "change value" << change_value <<std::endl;
        bc::wallet::payment_address change_address= std::get<2>(utxo_to_spend[0]);
        tx.outputs().push_back(createOutputP2PKH(change_address,change_value));
    }
    // If fees are greater than change, make change 0.
    else if (change_value - fees <= 0)
    {
        change_value = 0;
    }

    // Create output.
    tx.outputs().push_back(createOutputP2PKH(a_destinationAddress,a_satoshis));

    // Sign Transaction
    // bc::endorsement signature = create_signature(pub_key,privKey,tx)
    // bc::script = create_sig_script(signature,pub_key)
    // tx.inputs()[0].set_script(unlocking_script)

    // Return transaction.
    return tx;
};

/**
 * @brief Constructs P2PKH script transaction.
 *
 * @param a_publicKey, public key address of bitcoin payment address to use.
 * @param a_privKey
 * @param a_destinationAddress
 * @param a_satoshis
 * @param a_fees
 * @return true
 * @return false
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::chain::transaction
Transaction::P2PKH(bc::wallet::payment_address a_destinationAddress, unsigned long long
a_satoshis, unsigned long long a_fees)
{
```

```
unsigned long long input_value = 0;
unsigned long long change_value = 0;

// Start building Transaction.
// Instantiate the transaction object.
bc::chain::transaction tx = bc::chain::transaction();

// Set transaction version.
// uint32_t version = 1u;
// tx.set_version(version);

// Find unspent output.
m_utxo utxo_to_spend = unspent_output -> find_utxo(a_satoshis);

// Create inputs.
// For each input, point to unspent transaction output.
for( const auto &utxo : utxo_to_spend)
{
   // Create input.
   bc::chain::input input = bc::chain::input();
   bc::hash_digest utxo_hash = std::get<1>(utxo);
   bc::chain::output_point previous_output(utxo_hash,0);

   input.set_previous_output(previous_output);
   input.set_sequence(0xffffffff);
   tx.inputs().push_back(input);

   input_value += std::get<0>(utxo);

   m_last_utxo_address = std::get<2>(utxo);
};

change_value = input_value - a_satoshis;

// Find recommended fee.
int estimated_tx_bytes = calculateTxSize(tx.inputs().size(), 2);
unsigned long long fees = estimated_tx_bytes * a_fees;

// Subtract fees from the change.
if( change_value - fees > 0)
{
   change_value -= fees;
   std::cout << "fees to send: " <<fees << "change value" << change_value <<std::endl;
   bc::wallet::payment_address change_address= std::get<2>(utxo_to_spend[0]);
```

```
        tx.outputs().push_back(createOutputP2PKH(change_address,change_value));
    }
    // If fees are greater than change, make change 0.
    else if (change_value - fees <= 0)
    {
        change_value = 0;
    }

    // Create output.
    tx.outputs().push_back(createOutputP2PKH(a_destinationAddress,a_satoshis));

    // Return transaction.
    return tx;
};

/**
 * @brief Returns balanace of n payment address.
 *
 * @param a_address, payment address.
 * @return unsigned long long
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
unsigned long long
Transaction::getBalanceForAddress(bc::wallet::payment_address a_address)
{

    unsigned long long utxo = 0;
    bc::hash_digest utxo_hash;

    // Connect to libbitcoin servers.
    bc::client::obelisk_client &rpc = network->connect();

    // Lambda callback function for getting utxo for addy.
    static const auto on_done = [this, &utxo,&a_address,&utxo_hash](const
bc::chain::history::list& rows)
    {
        // For each row in chain history, check for balance.
        for(const auto& row: rows)
        {

            // Unspent transaction output.
            if (row.spend.hash() == bc::null_hash)
```

```
        {
            utxo += row.value;

            utxo_hash = row.output.hash();
            std::cout << bc::encode_hash(utxo_hash) << std::endl;
            unspent_output -> add_transaction(row.value, utxo_hash, a_address);

        }

        // Spent transaction output.

        {

            if(row.spend.hash() != bc::null_hash)
            {

m_transactions.push_back(std::make_tuple(row.value,row.spend.hash(),row.spend_height));
            }
            if(row.output.hash() != bc::null_hash)
            {

m_transactions.push_back(std::make_tuple(row.value,row.output.hash(),row.output_height));
            }

        }
    }
  };

  static const auto on_error = [](const bc::code ec)
  {
    Error::RecordError(std::string("Error connecting to bitcoin network."));
  };

  // Get Blockchain history on this address.
  rpc.blockchain_fetch_history3(on_error, on_done, a_address);

  // Wait for history to be fetched.
  rpc.wait();

  network -> disconnect();


  m_utxoMap[a_address] = std::make_pair(utxo_hash, utxo);
  // Return utxo for a_address.
```

```
    return utxo;
};

/**
 * @brief
 *
 * @param a_address, payment address to check UTXO for.
 * @param a_amount, minimum value of satoshis needed in UTXO.
 * @return bc::chain::points_value
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bc::chain::points_value
Transaction::getUTXOs(bc::wallet::payment_address a_address, unsigned long long a_amount)
{
    // Connect to libbitcoin servers.
    bc::client::obelisk_client &rpc = network->connect();

    bc::chain::points_value val1;
    static const auto on_done = [&val1](const bc::chain::points_value& vals) {

        std::cout << "Success: " << vals.value() << std::endl;
        val1 = vals;


    };

    static const auto on_error = [](const bc::code& ec) {

        std::cout << "Error Code: " << ec.message() << std::endl;

    };

    rpc.blockchain_fetch_unspent_outputs(on_error, on_done, a_address, a_amount,
bc::wallet::select_outputs::algorithm::greedy);

    rpc.wait();

    network -> disconnect();

    //return allPoints;
    return val1;
};
```

```
/**
 * @brief Broadcasts transaction to the network.
 *
 * @param tx
 * @return true
 * @return false
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bool
Transaction::broadcastTransaction(bc::chain::transaction tx)
{
   // Connect to libbitcoin servers.
   bc::client::obelisk_client &rpc = network->connect();

   static const auto on_done = [](const bc::code& ec) {

                    std::cout << "Success: " << ec.message() << std::endl;

           };

   static const auto on_error = [](const bc::code& ec) {

                    std::cout << "Error Code: " << ec.message() << std::endl;

           };

   rpc.transaction_pool_broadcast(on_error, on_done, tx);

   rpc.wait();

   network -> disconnect();

   return true;
};

/**
 * @brief Checks if the given payment address has recieved any bitcoin in its history.
 *
 * @param a_address, address to check.
 * @return true, payment address has recieved bitcoin.
 * @return false, payment address has never recieved bitcoin.
```

```
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bool
Transaction::isAddressUsed(bc::wallet::payment_address a_address)
{
   // Satoshis recieved.
   unsigned long long recieved = 0;

   // Connect to libbitcoin servers.
   bc::client::obelisk_client &rpc = network->connect();

   // Lambda callback function for getting utxo for addy.
   static const auto on_done = [&recieved](const bc::chain::history::list& rows)
   {
      // For each row in chain history, check for balance.
      for(const auto& row: rows)
      {
         recieved += row.value;
      }
   };

   static const auto on_error = [](const bc::code ec)
   {
      Error::RecordError(std::string("Error connecting to bitcoin network."));
   };

   // Get Blockchain history on this address.
   rpc.blockchain_fetch_history3(on_error, on_done, a_address);

   // Wait for history to be fetched.
   rpc.wait();

   network -> disconnect();

   // If address recieved any bitcoin, than it is used.
   if(recieved > 0)
   {
      return true;
   }
   else
   {
      return false;
```

```
    }

};

/**
 * @brief Calculates the balance of the wallet. Atlas calls this function until false is returned.
 *
 * @param a_address, payment address to check if address is used, and add any existing
balance.
 * @return true, address is used.
 * @return false, address is not used.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
bool
Transaction::calculateBalance(bc::wallet::payment_address a_address)
{
    std::cout << "Checking balance for " << a_address << std::endl;

    // Check if address is used.
    if(isAddressUsed(a_address))
    {
        // Get balance for the address. Add it to the sum.
        m_utxoSum += getBalanceForAddress(a_address);
        return true;
    }
    else
    {
        show_transaction_history();
        return false;
    }
};

/**
 * @brief Selector for current balance of utxo.
 *
 * @return unsigned long long
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
unsigned long long
Transaction::getBalance() const
```

```
{
   return m_utxoSum;
};

/**
 * @brief Get the transaction history.
 *
 * @return std::vector< m_tx >
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
std::vector< Transaction::m_tx >
Transaction::get_transaction_history() const
{
   return m_transactions;
};

/**
 * @brief Shows each transaction in transaction history.
 *
 * @author Philip Glazman
 * @date 4/28/18
 */
void
Transaction::show_transaction_history()
{
   std::sort(m_transactions.begin(),m_transactions.end(),compare_block_height);

   for(int i = 0; i < m_transactions.size(); i++)
   {
      std::cout << std::get<0>(m_transactions[i]) <<
bc::encode_hash(std::get<1>(m_transactions[i])) << std::get<2>(m_transactions[i]) << std::endl;
   }
};

/**
 * @brief Comparator function for comparing block height between two transactions. Used for
sorting.
 *
 * @param a
 * @param b
 * @return true
 * @return false
```

```
 *
 * @author Philip Glazman
 * @date 4/28/2018
 */
bool
Transaction::compare_block_height(const m_tx &a, const m_tx &b)
{
   return std::get<2>(a) > std::get<2>(b);
};

/**
 * @brief Get the last utxo address object
 *
 * @return bc::wallet::payment_address
 *
 * @author Philip Glazman
 * @date 4/29/18
 */
bc::wallet::payment_address
Transaction::get_last_utxo_address() const
{
   return m_last_utxo_address;
}
```

~~~~~~~~~~~~~~~~~~~~Source code for file valid_address.cpp~~~~~~~~~~~~~~~~~~~~

```
#include "valid_address.hpp"
#include <string>
#include "/usr/local/include/openssl/sha.h"

bool valid_address::valid(std::string a_address)
{
   unsigned char dec[32], d1[SHA256_DIGEST_LENGTH], d2[SHA256_DIGEST_LENGTH];

   return true;
}
```

~~~~~~~~~~~~~~~~~~~~Source code for file debug.cpp~~~~~~~~~~~~~~~~~~~~

```
#include "valid_address.hpp"
#include <iostream>

int main()
{
```

```
  std::cout << valid_address::valid("hello") << std::endl;
}
```

Test Cases

Launch new wallet.

Restore existing wallet using mnemonic phrase:

scatter	blanket
found	mother
issue	frequent
friend	acid
front	shaft
glare	loud

1. Sending a transaction to the address located on the main tab.

2. Check if the transaction was successfully broadcasted using

https://live.blockcypher.com/btc-testnet/

3. Test a simple bitcoin script:

    Witness: 1 2

    Witness Script: OP_ADD 3 OP_EQUAL

4. Compare the history of transactions in the history tab with the blockcypher explorer.