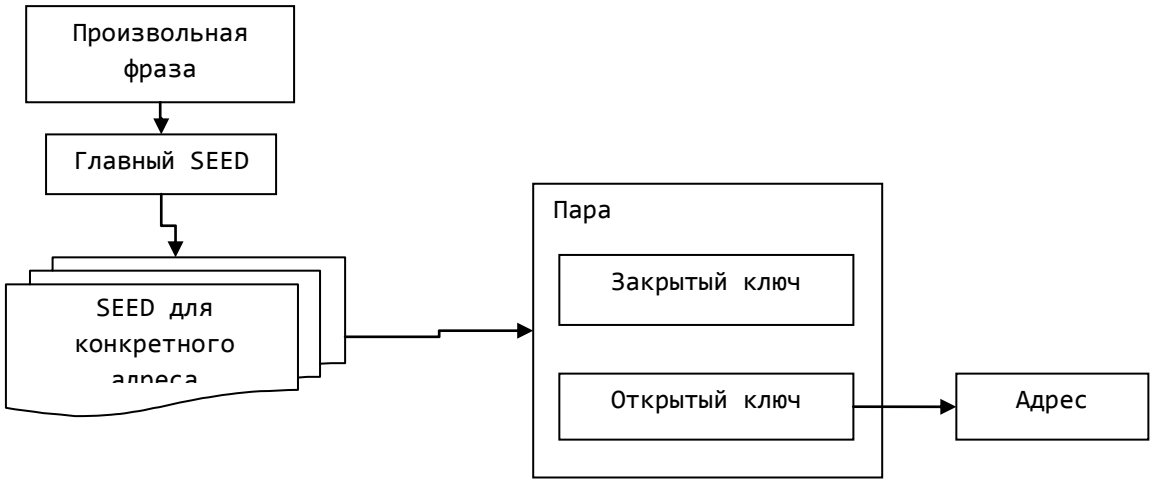
Описание библиотеки JS для работы с ERA

Все примеры даны для ознакомления.
Для работы системы необходимо запустить полную ноду системы ERA вашем компьютере. Как установить смотри в ReadMe1.docx

1. Создание адреса в системе ERA

В системе используется принцип шифрования с открытым ключом.
Общая схема формирования адреса приведена ниже.



Пример: createaddress.html

1.1. Для получения пары Закрытый ключ-Открытый ключ задается Произвольная фраза чем она длиннее, тем лучше или берется случайное число или комбинация любых символов. Главное условие – уникальность этой Фразы.

1.2. На основе Произвольной фразы вычисляем Главный Seed.
var seed = new Uint8Array(SHA256.digest(SHA256.digest(“[Произвольная фраза] ”)));

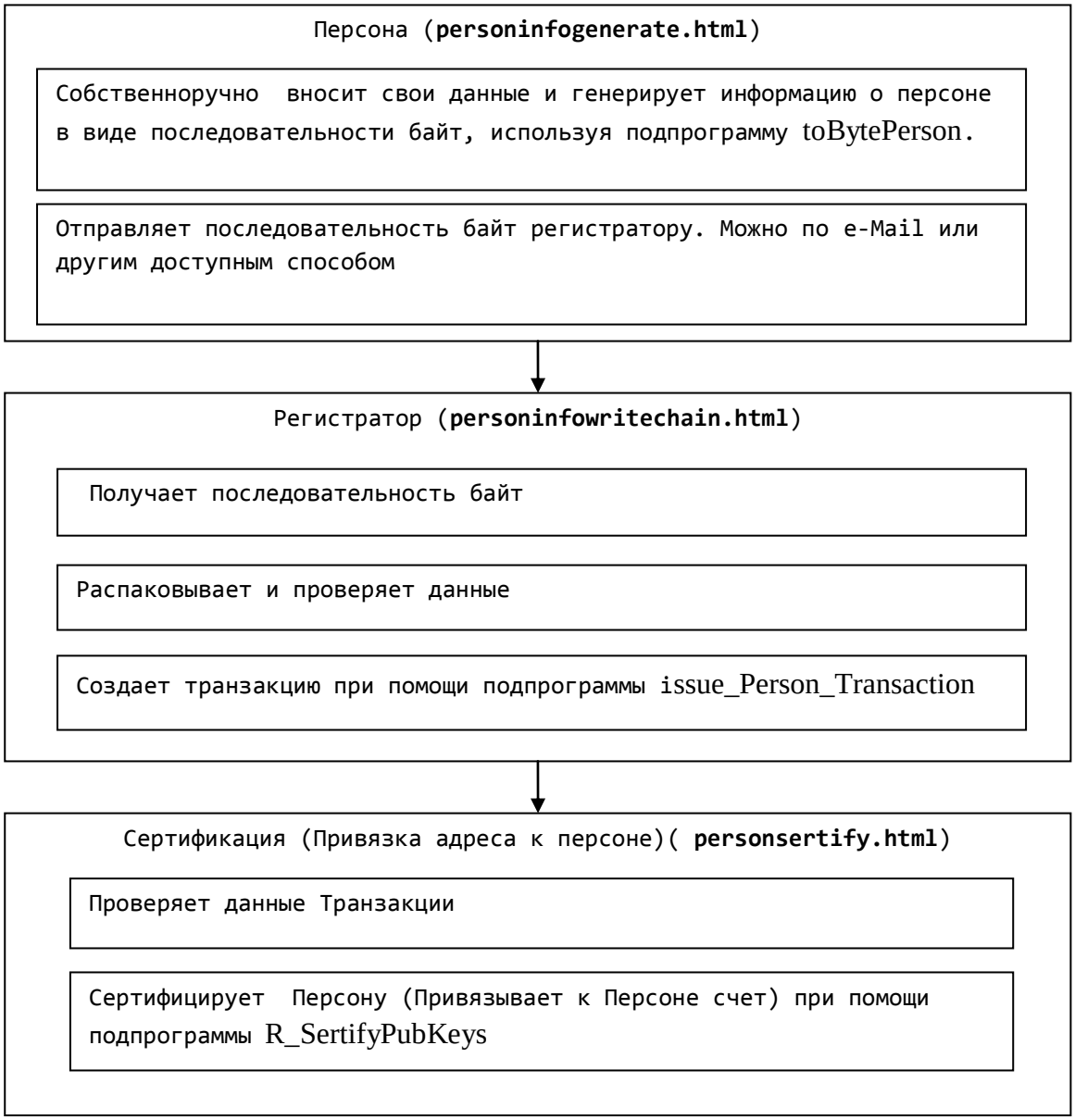
1.3. Далее вычисляем Seed для конкретного адреса
С одного Главного Seeda можно получить неограниченное количество счетов. Для этого к seedy добавляем приращение 0,1,2,…
Создадим Seed для конкретного адреса для приращения 0
var nonce =0;
var accountSeed = generateAccountSeed(seed, nonce, false);

1.4. Вычисляем пару Открытый ключ – Закрытый ключ
var keyPair = getKeyPairFromSeed(accountSeed);

1.5. Вычисляем Адрес для пары Открытый ключ – Закрытый ключ
var adderss = getAccountAddressFromPublicKey(keyPair.publicKey);

2. Работа с персонами

Система позволяет привязать конкретный адрес к персональным данным.
Для привязки адреса необходимо выполнить ряд действий.



Сертификацию может выполнить Регистратор. Персона и Регистратор разные люди.
Регистратором может быть только Сертифицированная Персона.

3. Создание последовательности байт транзакций

Файл: Пример: issueasset.html

3.1. Создать актив:

issue_Asset(keyPair, timestamp, asset_name, icon, image, description, quantity, scale, type_asset, port)

Возвращает byte[];

Параметры

keyPair	- pair public key + security key (byte64, byte32)
timestamp	- Unix timestamp. Формат длинное целое (Long). Пример 1514529622881
asset_name	- Наименование актива. Формат: Строка (String)
icon	- Иконка. Формат byte[]
image	- Картинка. Формат byte[]
description	- Описание. Формат: Строка
quantity	- количество. Формат: длинное целое
scale	- Точек после запятой. Формат: целое
type_asset*	- Тип актива
port	- ERA порт для боевой версии 9045 для отладочной 9065. Формат: целое.

Пример:

```
var raw = issue_Asset(keyPair, 1514529622881, "Asset asset develop", byte[], byte[], "description", 1000, 2, 0, 9045);
```

***** type_asset:

Значение	**Описание**
0	GOODS. Движимая вещь вовне - может быть доставлена и передана на хранение (товары)
1	ASSETS. * передача имущества не требует действий во вне - все исполняется тут же. Их можно дать в долг и заьрать самостоятельно * Требования не предъявляются. * цифровое имущество - не требует действий вовне и исполняется внутри платформы (токены, цифровые валюты, цифровые билеты, цифровые права и т.д.)
2	IMMOVABLE * передача в сосбтвенность, дать в аренду (по графику времени), взять на охрану * 1 : недвижимая вещь вовне - может быть передана в аренду (недвижимость)

11	outside CURRENCY * +++ деньги вовне - можно истребовать вернуть и подтвердить получение денег * === полный аналог OUTSIDE_CLAIM по действиям в протоколе - чисто для наименования другого
12	* outside SERVICE * +++ услуги во вне * === полный аналог OUTSIDE_CLAIM по действиям в протоколе - чисто для наименования другого
13	outside SHARE * +++ акция предприятия вовне * === полный аналог OUTSIDE_CLAIM по действиям в протоколе - чисто для наименования другого
14	outside BILL - вексель * +++ вексель на оплату во вне * === полный аналог OUTSIDE_CLAIM по действиям в протоколе - чисто для наименования другого
15	* outside BILL - вексель * +++ вексель на оплату во вне * === полный аналог OUTSIDE_CLAIM по действиям в протоколе - чисто для наименования другого
49	* outside CLAIMS * +++ требования и обязательства вовне - можно истребовать право и подтвердить его исполнение (ссуда, займ, услуга, право, требование, деньги, билеты и т.д.) * * учет обязательств прав и требований на услуги и действия во внешнем мире - в том числе займы, ссуды, кредиты, фьючерсы и т.д. * нельзя вернуть эмитенту - но можно потребовать исполнение прав и можно подтвердить исполнение (погасить требование) * это делается теми же трнзакциями что выдать и забрать долг у внутренних активов * И в момент погашения одновременно передается как имущество эмитенту
51	inside CURRENCY * +++ деньги * === полный аналог ASSET по действиям в протоколе - чисто для наименования другого
52	inside CLAIMS * +++ требования и обязательства * === полный аналог ASSET по действиям в протоколе - чисто для наименования другого
53	Inside Digital Share

54	inside BONUS * +++ бонусы - для анонимов так же платежи возможны * === ASSET - без обмена на бирже и можно анонимам переводить
55	inside RIGHTS * +++ права и доступы * === полный аналог ASSET по действиям в протоколе - чисто для наименования другого
56	inside VOTE * +++ права и доступы * === полный аналог ASSET по действиям в протоколе - чисто для наименования другого
119	inside CLAIMS * +++ требования и обязательства * === полный аналог ASSET по действиям в протоколе - чисто для наименования другого
123	ACCOUNTING * учетные единицы - нельзя на бирже торговать - они ничего не стоят, можно делать любые действия от своего имени * 4 : учетные единицы - не имеет стоимости и не может быть продано (бухгалтерский учет)

3.2. Переслать актив:

Файл: assetsend.html

generate_R_Send_TransactionBase(keyPair, recipient, asset_key, amount, scale, timestamp, title, message, enscript, is_text, port);

Возвращает byte[];

Параметры:

keyPair	- pair public key + security key (byte64, byte32)
recipient	- Адрес (adderss) получателя. Формат: byte[25].
asset_key	- Номер Актива. Формат: длинное целое
amount	- Сумма. Формат: Десятичное с точкой (Float). Может быть – null
scale	- Scale (Знаков после запятой)
timestamp	- Unix timestamp. Формат длинное целое
title	- Заголовок (String)
message	- Сообщение (String). Может быть - null
enscript	- 0 не зашифрованное; 1 зашифрованное
is_text	- 1
port	- ERA порт для боевой версии 9045 для отладочной 9065. Формат: целое.

Пример:

var recipient = Base58.decode(“7C64vWaRNvBwQK9YyyxRMZFHSJKgD7isUS”); // Addres ->byte[]
var raw = generate_R_Send_TransactionBase(keyPair, recipient, 1, 10.000001,8, 1514529622881, "Title", "Message", 0, 1, 9065);

3.3. Генерация данных для создания Персоны

Файл: personinfogenerate.html

toBytePerson(keyPair, name, icon, image, description, birthday, deathday, gender, race, birthLatitude, birthLongitude, skinColor, eyeColor, hairColor, height)

Возвращает byte[];

Параметры:

keyPair	- pair public key + security key (byte64, byte32)
name	- ИМЯ (String)
icon	- Icon (byte[])
image	- Image (byte[])
description	- Описание (String)
birthday	- День рождения. Unix timestamp (Long). 1514529622881
deathday	- День смерти. Unix timestamp (Long). 1514529622881
gender	- Пол. (0 - мужской, 1- женский)
race	- Расса. (String)
birthLatitude -	- Широта места рождения (float)
birthLongitude	- Долгота места рождения (float)
skinColor	- цвет кожи (String)
eyeColor	- цвет глаз (String)
hairColor	- цвет волос (String)
height	- высота (int)

Пример:
var rawPerson = toBytePerson(keyPair, "Person Name", byte[], byte[], "Person description", 1514529622881, 1514529622881, 0, "white", 31.2, 141.2, "white", "broun", "broun", 180);

3.4 Парсинг персоны

Парсит данные созданные функцией toBytePerson(…) пункт 3.3

Файл: personparseinfo.html

byteToPerson(raw)
raw - данные в формате Base58 созданные функцией toBytePerson(…) пункт 3.3

Возвращает массив с информацией о персоне.

Пример:

var person = byteToPerson(‘sdfsdfsdffsdfsdf’);

3.5. Создать персону

Файл: personinfowritechain.html

issue_Person_Transaction(keyPair, timestamp, data, port)

Возвращает byte[]

Параметры:

keyPair	- pair public key + security key (byte64, byte32)
timestamp	- Unix timestamp (Long). 1514529622881
data	- rawPerson
port	- ERA network PORT 9045 or dev:9065 (int)

Пример:
var raw =issue_Person_Transaction(keyPair, 1514529622881, byte[], 9045)

3.6. Сертифицировать персону. (Привязать адрес к персоне)

Файл: personcertify.html

R_SertifyPubKeys(keyPair, timestamp, person_key, publicKey, day, port);

Возвращает byte[]

Параметры:

keyPair	- pair public key + security key (byte64, byte32)
timestamp	- Unix timestamp (Long). 1514529622881
person_key	- Номер к которой привязывается адрес. (int)
publicKey	- Открытый ключ адреса, который привязывается к персоне. (byte[32])
day	- Количество дней на которое привязывается Адрес (int)
port	- ERA network PORT 9045 or dev:9065 (int)

Пример:

var publicKey =
Base58.decode("GAAt5zjMQbvGFCtyn3UCh2XzHwKgxh2LgHjHUn8qg7Ng");
var raw = R_SertifyPubKeys(keyPair, 1514529622881, 3, publicKey 365, 9045);

3.7. Подтверждение Транзакции

Файл: vouchtransaction.html

R_Vouch(keyPair, timestamp, blockHeigth, transNamber, port)

Возвращает byte[]

Параметры:
keyPair - pair public key + security key (byte64, byte32)
timestamp - Unix timestamp (Long). 1514529622881
blockHeigth - Номер блока (int)
transNamber - Номер транзакции в блоке (int)
port - ERA network PORT 9045 or dev:9065 (int)

Пример:
var raw = R_Vouch(keyPair, 1514529622881, 200, 1, 9045);

3.8. Создать статус

Файл: statusissue.html

issue_Status(keyPair, timestamp, name, icon, image, description, unique, port)

Возвращает byte[]

Параметры:
keyPair - pair public key + security key (byte64, byte32)
timestamp - Unix timestamp (Long). 1514529622881
name - name (String)
icon - Icon (byte[])
image - Image (byte[])
description - Description (String)
unique - Уникальный статус (0- уникальный, 1- множественный)
port - ERA network PORT 9045 or dev:9065 (int)

Пример:
var raw = issue_Status(keyPair, 1514529622881, "Status name", byte[], byte[], "Status description", 0, 9045);

3.9. Присвоить статус Итему

Файл: statusset.html

R_SetStatusToItem(keyPair, timestamp, status_Key, item_Type, item_Key, date_Start, date_End, value1, value2, string1, string2, refToParent, description , port){

Возвращает byte[]

Параметры:
keyPair - pair public key + security key (byte64, byte32)
timestamp - Unix timestamp (Long). 1514529622881
status_Key - Номер Статуса (long)
item_Type - Тип Итема (int)

ASSET_TYPE = 1;
IMPRINT_TYPE = 2;
NOTE_TYPE = 3;
PERSON_TYPE = 4;
STATUS_TYPE = 5;
UNION_TYPE = 6;

item_Key	- Item key (long)
date_Start	- Дата начала. Unix timestamp (Long). 1514529622881
date_End	- Дата окончания. Unix timestamp (Long). 1514529622881
value1	- Value %1. (int)
value2	- Value %2. (int)
string1	- String 3%. (String)
string2	- String 4%. (String)
refToParent	- Ссылка на другой статус. 0 – если нет ссылки (long)
description	- String 5%. (String)
port	- ERA network PORT 9045 or dev:9065 (int)

Пример:
var rew = R_SetStatusToItem(keyPair, 1514529622881, 1, 1, 1, 1514529622881, 1514529622881, 3, 10, "string1", "string2", 0, "description" , 9045)

3.10 Записать Hash

Файл: hashes_issue.html

write_Hashes(keyPair, timestamp, name, hashes, description, port)

Возвращает byte[]

Параметры:

keyPair	- pair public key + security key (byte64, byte32)
timestamp	- Unix timestamp (Long). 1514529622881
name	- name (String)
hashes[]	- Hashes[] (byte[32])
description	- Description (String)
port	- ERA network PORT 9046 or dev:9066 (int)

Приимер:
write_Hashes(keyPair, 1514529622881, "Haseshes name", hashes[], "Hashes description", 9046)

3.11 . Создать Ордер

Файл: ordercreate.html

Create_Order(keyPair, timestamp, have_asset, scale_h_asset , want_asset, scale_w_asset, have_ammount, want_ammount, port)

Возвращает byte[]

Параметры:

keyPair	- pair public key + security key (byte64, byte32)
timestamp	- Unix timestamp (Long). 1514529622881
have_asset	- key have Asset (Long)
scale_h_asset	- scale have Asset (Int) 0…16
want_asset	- key want Asset (Long)
scale_w_asset	-scale want Asset(Int) 0…16
have_ammount	- have amount (Float) 100.23
want_ammount	- want amount (Float) 123.456
port	- ERA network PORT 9045 or dev:9065 (int)

Пример:

```
var raw = Create_Order(keyPair, 1514529622881, 1,8, 2,8, 100.23, 123.456, 9045);
```

3.12. Отменить Ордер

Файл: orderdelete.html

Cancel_Order(keyPair, timestamp, order_sign, port)

Возвращает byte[]

Параметры:

keyPair	- pair public key + security key (byte64, byte32)
timestamp	- Unix timestamp (Long). 1514529622881
order_sign	- Signature ордера который надо удалить (byte[])
port	- ERA network PORT 9045 or dev:9065 (int)

Пример:

```
var  raw = Cancel_Order(keyPair, 1514529622881, byte[], 9045)
```

3.13 . Документ

4. Отправка данных на сервер

Сервер принимает, как GET так и POST, HTTP запросы.
Если, данные имеют большую длину, отправляйте POST запросом. В любом случае POST предпочтителен.

4.1. GET запрос из адресной строки браузера

http://127.0.0.1:WEB_PORT/api/broadcast/{raw}

4.2. GET запрос при помощи JQuery

```
$.get( http://127.0.0.1: +WEB_PORT + "/api/broadcast/"+ Base58.encode(raw) , function( data )
{

                $("#output").val("Result: " + data.status +"       Message:" +
data.message);
        })
        .fail(function() {
                $("#output").val('error!');
        });
```

4.3. POST запрос

```
        $.post(http://127.0.0.1: +WEB_PORT + "/api/broadcast", "raw=" +Base58.encode(raw) ,
function( data) {

                $("#output").val("Result: " + data.status +"       Message:" +
data.message);

        })
        .fail(function() {
                $("#output").val('error!');
        });
```

Где:
WEB_PORT – web порт. Отличен от порта сети. посмотреть/ изменить WEB_PORT можно в основной ноде. Меню->файл->настройки
raw – последовательность байт полученная при помощи функций приведенных в предыдущей главе. За исключением функции toBytePerson()
$("#output") – элемент в который выводим ответ от сервера

4.3.Список ответов
VALIDATE_OK = 1;
INVALID_MAKER_ADDRESS = 5;
INVALID_REFERENCE = 6;
INVALID_TIMESTAMP = 7;
INVALID_ADDRESS = 8;
INVALID_FEE_POWER = 9;
NOT_ENOUGH_FEE = 10;
NO_BALANCE = 11;
INVALID_PUBLIC_KEY = 12;
INVALID_RAW_DATA = 13;
INVALID_DATE = 14;
INVALID_CREATOR = 15; // for some reasons that creator is invalid (same as trade order)
INVALID_SIGNATURE = 16;
NO_DEBT_BALANCE = 17;
NO_HOLD_BALANCE = 18;
NOT_ENOUGH_RIGHTS = 20;
OWNER_NOT_PERSONALIZED = 21;
ACCOUNT_ALREADY_PERSONALIZED = 23;
TRANSACTION_DOES_NOT_EXIST = 24;
CREATOR_NOT_PERSONALIZED = 25;
RECEIVER_NOT_PERSONALIZED = 26;

ASSETS:
INVALID_QUANTITY = 30;
NEGATIVE_AMOUNT = 32;
INVALID_AMOUNT = 33;
INVALID_RETURN = 34;
HAVE_EQUALS_WANT = 35;
ORDER_DOES_NOT_EXIST = 36;
INVALID_ORDER_CREATOR = 37;
INVALID_PAYMENTS_LENGTH = 38;
NEGATIVE_PRICE = 39;
INVALID_PRICE = 40;
INVALID_CREATION_BYTES = 41;
INVALID_TAGS_LENGTH = 42;
INVALID_TYPE_LENGTH = 43;
NOT_MOVABLE_ASSET = 44;
NOT_DEBT_ASSET = 45;
INVALID_NAME_LENGTH = 50;
INVALID_ICON_LENGTH = 51;
INVALID_IMAGE_LENGTH = 52;
INVALID_DESCRIPTION_LENGTH = 53;
INVALID_VALUE_LENGTH = 55;

POLL;
INVALID_OPTIONS_LENGTH = 80;
INVALID_OPTION_LENGTH = 81;
DUPLICATE_OPTION = 82;

POLL_ALREADY_CREATED = 83;
POLL_ALREADY_HAS_VOTES = 84;
POLL_NOT_EXISTS = 85;
POLL_OPTION_NOT_EXISTS = 86;
ALREADY_VOTED_FOR_THAT_OPTION = 87;

INVALID_DATA_LENGTH = 88;
INVALID_DATA = 89;
INVALID_PARAMS_LENGTH = 90;
INVALID_URL_LENGTH = 91;
INVALID_HEAD_LENGTH = 92;

ITEMS
INVALID_ITEM_VALUE = 100;
ITEM_DOES_NOT_EXIST = 101;
ITEM_ASSET_NOT_EXIST = 102;
ITEM_IMPRINT_DOES_NOT_EXIST = 103;
ITEM_NOTE_NOT_EXIST = 104;
ITEM_PERSON_NOT_EXIST = 105;
ITEM_STATUS_NOT_EXIST = 106;
ITEM_UNION_NOT_EXIST = 107;
ITEM_DOES_NOT_STATUSED = 108;
ITEM_DOES_NOT_UNITED = 109;
ITEM_DUPLICATE_KEY = 110;
ITEM_DUPLICATE = 111;
AMOUNT_DIVISIBLE = 115;
ITEM_PERSON_LATITUDE_ERROR = 120;
ITEM_PERSON_LONGITUDE_ERROR = 121;
ITEM_PERSON_RACE_ERROR = 122;
ITEM_PERSON_GENDER_ERROR = 123;
ITEM_PERSON_SKIN_COLOR_ERROR = 124;
ITEM_PERSON_EYE_COLOR_ERROR = 125;
ITEM_PERSON_HAIR_COLOR_ERROR = 126;
ITEM_PERSON_HEIGHT_ERROR = 127;
ITEM_PERSON_OWNER_SIGNATURE_INVALID = 128;
INVALID_UPDATE_VALUE = 140;
INVALID_TRANSACTION_TYPE = 150;
INVALID_BLOCK_HEIGHT = 200;
INVALID_BLOCK_TRANS_SEQ_ERROR = 201;
NOT_YET_RELEASED = 299;
AT_ERROR = 300;

Приложение А
Описание структуры данных

Send Asset

часть	Наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[31,0,0,0] *
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	RECIPIENT	BYTE[25]	BYTE[25]	
8	ASSET KEY	BYTE[8]	Long	
9	AMMOUNT	BYTE[8]	Big Decimal	
10	HEAD LENGHT	BYTE[1]	Int	
11	HEAD	BYTE[HEAD LENGTH]	String	
12	MESSAGE LENGTH	BYTE[4]	long	
13	MESSAGE	BYTE[MESSAGE LENGTH]	String	
14	ENSCRYPTED	BYTE[1]	int	
15	IS TEXT	BYTE[1]	int	

Signature

1	TYPE_TRANSACTION	BYTE[4]		[31,0,0,0] *
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	RECIPIENT	BYTE[25]	BYTE[25]	
7	ASSET KEY	BYTE[8]	Long	
8	AMMOUNT	BYTE[8]	Big Decimal	
9	HEAD LENGHT	BYTE[1]	Int	
10	HEAD	BYTE[HEAD LENGTH]	String	
11	MESSAGE LENGTH	BYTE[4]	int	

12	MESSAGE	BYTE[MESSAGE LENGTH]	String	
13	ENSCRYPTED	BYTE[1]	int	
14	IS TEXT	BYTE[1]	int	
15	PORT	BYTE[4]	int	

* TYPE_TRANSACTION состоит из 4 байт каждый байт заполняется определенным образом:
TYPE_TRANSACTION [0] - record type равен 31
TYPE_TRANSACTION [1] - record version {0,1,2}
TYPE_TRANSACTION [2] - property 1
TYPE_TRANSACTION [3] - property 2

version 0
TYPE_TRANSACTION [1] =0
TYPE_TRANSACTION [2] = -128 if NO AMOUNT
TYPE_TRANSACTION [3] = -128 if NO DATA

version 1
TYPE_TRANSACTION [1] = 1
TYPE_TRANSACTION [1] (version) = 1 - if backward - CONFISCATE CREDIT

version 2
TYPE_TRANSACTION [1] =2

PROPERTY 1:
TYPE_TRANSACTION [2].0 = -128 if NO AMOUNT
TYPE_TRANSACTION [2].1 = -64 if backward - CONFISCATE CREDIT

PROPERTY 2:
TYPE_TRANSACTION [3].0 = -128 if NO DATA

version 3
TYPE_TRANSACTION [1] =3

PROPERTY 1:
TYPE_TRANSACTION [2].0 = -128 if NO AMOUNT - check sign
TYPE_TRANSACTION [2].1 = 64 if backward (CONFISCATE CREDIT, ...)

PROPERTY 2:
TYPE_TRANSACTION [3].0 = -128 if NO DATA - check sign = '10000000' = Integer.toBinaryString(128) - assertEquals((byte)128, (byte)-128);
TYPE_TRANSACTION [3].3-7 = point accuracy: -16..16 = BYTE - 16

Создать актив

	наименование	Формат	Приводим к виду	Значение по умолчанию
1	TYPE_TRANSACTION	BYTE[4]		[21,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	TYPE ITEM	BYTE[2]	int	[2,0] второй параметр Movable (0 - not movable, 1- movable)
8	OWNER	BYTE[32]	BYTE[32]	
9	NAME LENGTH	BYTE[1]	int	
10	NAME	BYTE[NAME LENGTH]	String UTF-8	
11	ICON LENGTH	BYTE[4]	int	
12	ICON	BYTE[ICON LENGTH]	???	
13	IMAGE LENGTH	BYTE[4]	int	
14	IMAGE	BYTE[IMAGE LENGTH]	???	
15	DESCRIPTION LENGTH	BYTE[4]	int	
16	DESCRIPTION	BYTE[DESCRIPTION LENGTH	String	
17	QUANTITY	BYTE[8]	Long	
18	SCALE	BYTE[1]	int	
19	TYPE ASSET	BYTE[1]	int	

Signature

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[21,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	TYPE ITEM	BYTE[2]	int	[2,0]
7	OWNER	BYTE[32]	BYTE[32]	
8	NAME LENGTH	BYTE[1]	int	
9	NAME	BYTE[NAME LENGTH]	String UTF-8	
10	ICON LENGTH	BYTE[4]	int	

11	ICON	BYTE[ICON LENGTH]	???	
12	IMAGE LENGTH	BYTE[4]	int	
13	IMAGE	BYTE[IMAGE LENGTH]	???	
14	DESCRIPTION LENGTH	BYTE[4]	int	
15	DESCRIPTION	BYTE[DESCRIPTION LENGTH	String	
16	QUANTITY	BYTE[8]	Long	
17	SCALE	BYTE[1]	int	
18	TYPE ASSET	BYTE[1]	int	
15	PORT	BYTE[4]	int	

Создание инфо Person

	наименование	Формат	Приводим к виду	
1	TYPE ITEM	BYTE[1]	int	[1,1]
2	OWNER	BYTE[32]	BYTE[32]	
3	NAME LENGTH	BYTE[1]	int	
4	NAME	BYTE[NAME LENGTH]	String UTF-8	
5	ICON LENGTH	BYTE[4]	int	
6	ICON	BYTE[ICON LENGTH]	???	
7	IMAGE LENGTH	BYTE[4]	int	
8	IMAGE	BYTE[IMAGE LENGTH]	???	
9	DESCRIPTION LENGTH	BYTE[4]	int	
10	DESCRIPTION	BYTE[DESCRIPTION LENGTH	String	
11	birthdayBytes	BYTE[8]	Long	
12	deathdayBytes	BYTE[8]	Long	
13	Gender	BYTE[1]	int	
14	raceLength	BYTE[1]	int	
15	raceBytes	BYTE[raceLength]	String	
16	birthLatitude	BYTE[4]	float	
17	birthLongitude	BYTE[4]	float	
18	skinColorLength	BYTE[1]	int	
19	skinColor	BYTE[skinColorLength]	string	
20	eyeColorLength	BYTE[1]	int	
21	eyeColorBytes	BYTE[eyeColorLength]	string	
22	hairColorLength	BYTE[1]	int	
23	hairColorBytes	BYTE[hairColorLength]	string	
24	Height	BYTE[1]	int	
25	ownerSignature	BYTE[64]	BYTE[]	

Issue Person Transaction

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[24,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	инфо Person	Все данные из инфо Person		

Signature Person Transaction

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[24,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	инфо Person	Все данные из инфо Person		
7	PORT	BYTE[4]	int	

Sertify Person Transaction

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[36,0,1,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	PERSON KEY	BYTE[8]	Long	
8	Account PUBLIC KEY	BYTE[32]	BYTE[32]	
9	DAY	BYTE[4]	int	

Signature

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[36,0,1,0] 1-один адрес привязывается, можно привязать до

				256
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	PERSON KEY	BYTE[8]	Long	
7	Account PUBLIC KEY	BYTE[32]	BYTE[32]	
8	DAY	BYTE[4]	int	
9	PORT	BYTE[4]	int	

R_Vouch

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[40,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	BLOCK_HEIGTH	BYTE[4]	int	
8	TRANCACTION NAMBER IN BLOCK	BYTE[4]	int	

Signature

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[40,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	BLOCK_HEIGTH	BYTE[4]	int	
7	TRANCACTION NAMBER IN BLOCK	BYTE[4]	int	
8	PORT	BYTE[4]	int	

Issue Status

	наименование	Формат	Приводим к виду	Значение по умолчанию
1	TYPE_TRANSACTION	BYTE[4]		[25,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]

6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	TYPE ITEM	BYTE[2]	int	[1,X]; X=1 уникальный статус; X =0 Не уникальный
8	OWNER	BYTE[32]	BYTE[32]	
9	NAME LENGTH	BYTE[1]	int	
10	NAME	BYTE[NAME LENGTH]	String UTF-8	
11	ICON LENGTH	BYTE[4]	int	
12	ICON	BYTE[ICON LENGTH]	???	
13	IMAGE LENGTH	BYTE[4]	int	
14	IMAGE	BYTE[IMAGE LENGTH]	???	
15	DESCRIPTION LENGTH	BYTE[4]	int	
16	DESCRIPTION	BYTE[DESCRIPTION LENGTH	String	

Signature

	наименование	Формат	Приводим к виду	Значение по умолчанию
1	TYPE_TRANSACTION	BYTE[4]		[25,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
7	TYPE ITEM	BYTE[2]	int	
8	OWNER	BYTE[32]	BYTE[32]	[1,X]; X=1 уникальный статус; X =0 Не уникальный
9	NAME LENGTH	BYTE[1]	int	
10	NAME	BYTE[NAME LENGTH]	String UTF-8	
11	ICON LENGTH	BYTE[4]	int	
12	ICON	BYTE[ICON LENGTH]	???	
13	IMAGE LENGTH	BYTE[4]	int	
14	IMAGE	BYTE[IMAGE LENGTH]	???	
15	DESCRIPTION LENGTH	BYTE[4]	int	
16	DESCRIPTION	BYTE[DESCRIPTION LENGTH	String	

17	PORT	BYTE[4]	int	

R_SetStatusToItem Transaction

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[37,0,0,Params]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	KEY STATUS	BYTE[8]	Long	
9	ITEM TYPE	BYTE[1]	Int	
8	KEY ITEM	BYTE[8]	Long	
10	DATA START	BYTE[8]	Long	
11	DATA END	BYTE[8]	Long	
12	VALUE1	BYTE[8]	Long	
13	VALUE2	BYTE[8]	Long	
14	DATA1 LENGTH	BYTE[1]	INT	
15	DATA	BYTE[DATA1 LENGTH]	STRING	
16	DATA2 LENGTH	BYTE[1]	INT	
17	DATA	BYTE[DATA2 LENGTH]	STRING	
18	REF TO PARENT	BYTE[8]	Long	
19	DESCRIPTION	BYTE[]	STRING	

SIGNATURE

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[37,0,0,Params]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	KEY STATUS	BYTE[8]	Long	
7	ITEM TYPE	BYTE[1]	Int	
9	KEY ITEM	BYTE[8]	Long	
8	DATA START	BYTE[8]	Long	
10	DATA END	BYTE[8]	Long	
11	VALUE1	BYTE[8]	Long	
12	VALUE2	BYTE[8]	Long	
13	DATA1 LENGTH	BYTE[1]	INT	
14	DATA	BYTE[DATA1 LENGTH]	STRING	
15	DATA2 LENGTH	BYTE[1]	INT	

16	DATA	BYTE[DATA2 LENGTH]	STRING	
17	REF TO PARENT	BYTE[8]	Long	
18	DESCRIPTION	BYTE[]	STRING	
19	PORT	BYTE[4]	int	

Create_Order

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[50,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	HAVE_ASSET	BYTE[8]	Long	Код Актива, который имеем
9	WANT_ASSET	BYTE[8]	Long	Код Актива, который хотим
8	HAVE_AMOUNT	BYTE[8]	Big Decimal	Кол-во, которое имеем
10	WANT_AMOUNT	BYTE[8]	Big Decimal	Кол-во, которое хотим

Signature

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[50,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	HAVE_ASSET	BYTE[8]	Long	
7	WANT_ASSET	BYTE[8]	Long	
9	HAVE_AMOUNT	BYTE[8]	Long	
8	WANT_AMOUNT	BYTE[8]	Long	
10	PORT	BYTE[4]	int	

Cancel Order

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[51,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]

6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	SIGNATUPE ORDER	BYTE[64]	BYTE[64]	

Signature

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[51,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE ORDER	BYTE[64]	BYTE[64]	
7	PORT	BYTE[4]	int	

R Hash

часть	Наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[41,0,0,0] Третий и четвертый байт – кол-во хешей в транзакции
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	URL LENGTH	BYTE[1]	Integer	
8	URL	BYTE[URL LENGTH]	String	
9	DATA_SIZE_LENGTH	BYTE[4]	Integer	
10	DATA	BYTE[DATA_SIZE_LENGTH]	String	
11	Hashes[i]	BYTE[32][i]	BYTE[32]	i – количество записываемых HASH . Длина hash – 32 байта

Signature

1	TYPE_TRANSACTION	BYTE[4]		[41,0,0,0]
2	TIMESTAMP	BYTE[8]	Long	

3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	URL LENGTH	BYTE[1]	Integer	
7	URL	BYTE[URL LENGTH]	String	
8	DATA_SIZE_LENGTH	BYTE[4]	Integer	
9	DATA	BYTE[DATA_SIZE_LENGTH]	String	
10	Hashes[i]	BYTE[32][i]	BYTE[32]	i – количество записываемых HASH. Длина hash – 32 байта
11	PORT	BYTE[4]	int	

Document

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[35,0,0,0]; Если параметр[3] < 0, то присутствует шаблон. Первые 2 бита указывают количество размера байт количества подписавших документ.???? Если параметр[4] < 0, то присутствуют данные.
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	SIGNATUPE	BYTE[64]	BYTE[64]	
7	TEMPLATE KEY	BYTE[8]	Long	Ключ шаблона

				Читается/Пишется только, если параметр[3] < 0
8	DATA SIZE	BYTE[4]	Integer	Размер данных Читается/Пишется только, если параметр[4] < 0
9	DATE	BYTE[DATA SIZE]	BYTE[]	Данные Читается/Пишется только, если параметр[4] < 0
10	ENSCRYPTED	BYTE[1]	int	Читается/Пишется только, если параметр[4] < 0
11	IS TEXT	BYTE[1]	int	Читается/Пишется только, если параметр[4] < 0
12	signersLenBytes	BYTE[1]	int	???? по умолчанию отсутствует
13	Signers[i]	BYTE[32]	BYTE[]	???? по умолчанию отсутствует
14	signatures[i]	BYTE[64]	BYTE[]	???? по умолчанию отсутствует

Структура данных Документа (DATE)

	наименование	Формат	Приводим к виду	
1	DATA VERSION	BYTE[6]	String	По умолчанию "v 2.00"
2	TITLE SIZE	BYTE[4]	int	
3	TITLE	BYTE[TITLE SIZE]	String	
4	JSON SIZE	BYTE[4]	int	
5	JSON	BYTE[JSON SIZE]	String	
6	FILES	BYTE[FILES][]	BYTE[]	Данные прикрепленных файлов (если имеются)

Структура JSON

```
{
"TM":"1001",
"PR":{
       "date":"21\/02\/2018",
```

```
        "id":"988"
},
"HS":{
        "GgEUSYQ5yqdTnGoNSPqn8ScP7jjjvj9nxmHz9N9gGzwq":"from file
C:\\Users\\Саша\\Desktop\\2222.era",
        "DKr7eqCAps3GaDo1EjberQDtMwZ2AorBzbkfTC4X3PYf":"from file
C:\\Users\\Саша\\Desktop\\Daemon_Pro.rar",
        "CX9bnpSb4vevDTRCfD93TsMzs5kqQ1PXVfG3Nr2h6rGv":"from file
C:\\Users\\Саша\\Desktop\\bp.exe - Ярлык.lnk"
},
"MS":"сообщение",
"F":{
        "0":{
                "ZP":"false",
                "SZ":"560",
                "FN":"3333.json"
        },
        "1":{
                "ZP":"false",
                "SZ":"123",
                "FN":"2222.era"
        }
}
}
```


Signature

	наименование	Формат	Приводим к виду	
1	TYPE_TRANSACTION	BYTE[4]		[35,0,0,0]; Если параметр[3] < 0, то присутствует шаблон. Первые 2 бита указывают количество размера байт количества подписавших документ.???? Если параметр[4] < 0, то присутствуют данные.
2	TIMESTAMP	BYTE[8]	Long	
3	REFERENCE	BYTE[8]	Long	[0,0,0,0,0,0,0,0]
4	CREATOR PUBLIC KEY	BYTE[32]	BYTE[32]	
5	FEE POW	BYTE[1]	BYTE[1]	[0]
6	DATA SIZE	BYTE[4]	Int	

7	DATA	BYTE[DATA SIZE]	BYTE[0]	
8	ENSCRYPTED	BYTE[1]	int	0
9	IS TEXT	BYTE[1]	int	1
10	PORT	BYTE[4]	int	