

2018年2月5日

第1回 技術応用部会

Bitcoin (Testnet) に触ないでみよう

BCCC 技術応用部会長
インフォテリア株式会社
森 一弥

今回の趣旨と目的



- 初回なのでできるだけ簡単に！
 - ✦ 使えるものはなんでも使う
 - ✦ テキストエディタだけでなんとかしたい
- ブロックチェーンプログラムを「やった感」を大事に！
 - ✦ 環境構築で時間が終わっちゃうセミナーではつまんないですよね？
 - ✦ とにかく「送金」と「残高確認」してみましょう
- 続きは自分でできるように！
 - ✦ リファレンスとかを見れば自分でもできるもので



そこそこメジャー（だと思う）なライブラリ「**BitCore**」と
HTML + JavaScript (**JQuery**) を使ってみます。

アジェンダ



- ライブラリを手に入れる
- アカウントをつくろう
- Bitcoinを手に入れる！
- ブロックチェーンエクスプローラでみてみよう
- プログラムから残高照会
- 送金してみよう
- ブロックチェーンにラクガキしてみよう
- まとめ

ご注意！



- 日進月歩の技術ですので、しばらくすると使えなくなったり、時代遅れになることも予想されます。
- 「技術者」なら今回の基礎情報を自ら補完し、さらなる活用を目指してください！！

ライブラリを手に入れる

ライブラリを手に入れる①



- 入れてない人は、Node.js をインストールしましょう
- <https://nodejs.org/ja/>
- 入れたことない人は推奨版を入れれば良いんじゃないですかね



ライブラリを手に入れる②



- 以下のコマンドを「ターミナル」で実行してバージョンが出ればOKです。

```
node -v
```

Windowの場合は、スタートメニューにNode.jsのコマンドプロンプトが追加されるようなので、そちらで

- 最近のmacだと、必要なコマンドラインツールが消えてるらしいので、以下のコマンドを一応実行したほうが良いかもしれません。

```
xcode-select --install
```

Macの場合のみ！

ライブラリを手に入れる③



- 適当なディレクトリを作って、移動しましょう。

```
mkdir bccc  
cd bccc
```

- Node.jsのパッケージファイルを作ります。

```
npm init --y
```


ライブラリを手に入れる④



- Node.jsのライブラリをブラウザで使うように変換できるツール「Browserify」をインストールします。

```
npm install -g browserify
```

- Node.jsのコード圧縮ツール「uglifyjs」をインストールします。

```
npm install -g uglify-js
```

ライブラリを手に入れる⑤



- BitCoreをインストールします。

```
npm install bitcore-lib
```

うまくいくと、「node_modules」フォルダができて、「bitcore-lib」が入るはずです。

- 「Browserify」でブラウザ用のライブラリを生成します。

```
browserify --require ./node_modules/bitcore-lib/index.js:bitcore-lib | uglifyjs > ./bitcore-lib.min.js
```

資料では2行にしてますが、
コマンドは全て1行で！

アカウントをつくらう

アカウントを作ろう① ～ 準備 ～



- 先程作った適当なフォルダにHTMLファイルを新規作成して、以下のような感じでjQuery と Bitcore を使えるようにしましょう。

account.html

```
<!DOCTYPE html>
<html lang="jp">
  <head>
    <meta charset="utf-8">
    <title>アカウント作成</title>
    <script type="text/javascript" src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
    <script type="text/javascript" src="bitcore-lib.min.js"></script>
  </head>
  <body>
    <div id="contents"></div>
    <script type="text/javascript" src="account.js"></script>
  </body>
</html>
```

文字コードは環境に合わせて！

取得したライブラリ

これから作るJSファイル

アカウントを作ろう② ～ つくる ～



- HTMLと同じフォルダにJSファイルを新規作成してスクリプトを書いていきましょう。

account.js

```
//ライブラリを読み込み
var bitcore = require('bitcore-lib');

//秘密鍵の新規作成
var privateKey = new bitcore.PrivateKey('testnet');

//公開鍵を秘密鍵から生成
var publicKey = privateKey.toPublicKey();

//アドレスを秘密鍵から生成
var address = privateKey.toAddress('testnet');
```

アカウントを作ろう③ ～ 表示 ～



- 作成した秘密鍵、公開鍵、アドレスをHTMLに表示させましょう。

account.js (前ページの続き)

```
$("#contents").append("<div>秘密鍵 : " + privateKey + "</div>");
```

```
$("#contents").append("<div>公開鍵 : " + publicKey + "</div>");
```

```
$("#contents").append("<div>アドレス : " + address + "</div>");
```

アカウントを作ろう④ ～ 実行 ～



- 作成したHTMLとJSファイルを保存して、ブラウザでHTMLを実行してみましょう。
- 表示内容はこの後も使いますので、コピペするなりしてどこかに保存してください。



Bitcoinを手に入れる！

Bitcoinを手に入れる



- なんとTestnet用のBitcoinは、配ってくれている方がいらっしやるのです！
- 「bitcoin testnet faucet」でインターネット検索してみてください。

bitcoin testnet faucet



- 資料作成時では、以下のサイトでご自身で先程作ったBitcoinアドレスを入力するともらえました！

★ <https://testnet.manu.backend.hamburg/faucet>

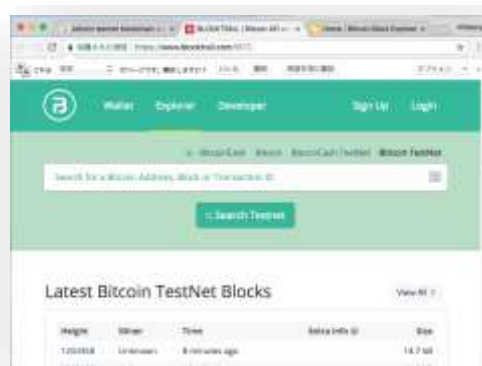
ブロックチェーン
エクスプローラで
みてみよう

ブロックチェーンエクスプローラって？



- 「ブロックチェーンエクスプローラー」とは、その名の通り、ブロックチェーンのトランザクションなどを見られるものです。
- 「bitcoin testnet blockchain explorer」で検索すればいくつか出てくるはずです。

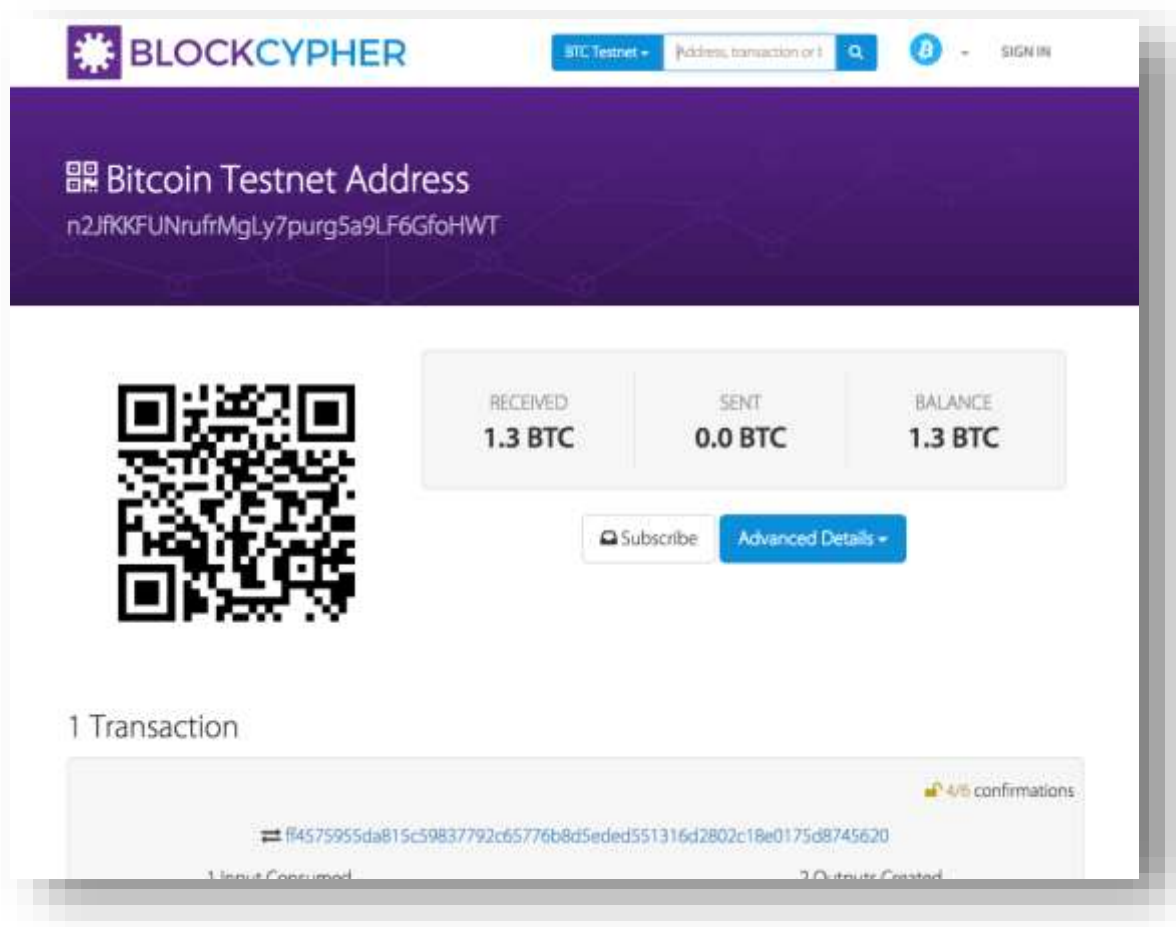
bitcoin testnet blockchain explorer



自分のアドレスを検索



- たぶん、どのBlockchain Explorerでもアドレスの検索ができるはずです。
- ご自分の「アドレス」を検索してみてください。



トランザクションをよく見よう



1 Transaction

ff4575955da815c59837792c65776b8d5eded551316d2802c18e0175d8745620

1 Input Consumed

864.66959087 BTC from
2N6bP5aYvwAHazyUHt27uCpyQoPbyZ...

2 Outputs Created

1.3 BTC to
n2JfKKFUNrufrMgLy7purg5a9LF6GfoH...

863.36859087 BTC to
2N9GJJJoXJ37LsAimD7NLbdwjEC4iCBz...

Value Transacted : 864.66859087 BTC

自分のアドレスに送られたBTC

送金元である「Faucet」
の人のアドレス

送り主に戻された、
「おつり」

プログラムから 残高照会

RESTで取得する！



- プログラムから、ブロックチェーンエクスプローラーと同じ情報が取れば、残高も取れそう！
- 自分でBitCoreの「Insight」というブロックチェーンエクスプローラーを立てるのが正規の手順ですが、今回は端折ります！
- Testnet用のInsight APIも公開してくれているサイトが幾つかあります。
- RESTで取得するのが簡単そうですね！

残高照会① ～ 準備 ～



- アカウント作成と同様にHTMLファイルを新規作成して、以下のような感じでjQuery と Bitcore を使えるようにしましょう。

balance.html

```
<!DOCTYPE html>
<html lang="jp">
  <head>
    <meta charset="utf-8">
    <title>残高照会</title>
    <script type="text/javascript" src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
  </head>
  <body>
    <pre id="contents"></pre>
    <script type="text/javascript" src="balance.js"></script>
  </body>
</html>
```

文字コードは環境に合わせて！

Preタグにしています

これから作るJSファイル

残高照会② ～ つくる ～



- HTMLと同じフォルダにJSファイルを新規作成してスクリプトを書いていきましょう。

balance.js

```
var endpoint = "https://testnet.blockexplorer.com";  
var address = "取得したアドレス";  
  
$.ajax({  
  type: "get",  
  url: endpoint + "/api/addr/" + address ,  
  dataType: 'json'  
}).done(function(response){  
  $("#contents").text(JSON.stringify(response,null,"¥t"));  
}).fail(function(errordata){  
  console.log(errordata);  
});
```

エンドポイントは、以下の2つはいけ
そうです。

<https://testnet.blockexplorer.com>
<https://test-insight.bitpay.com>

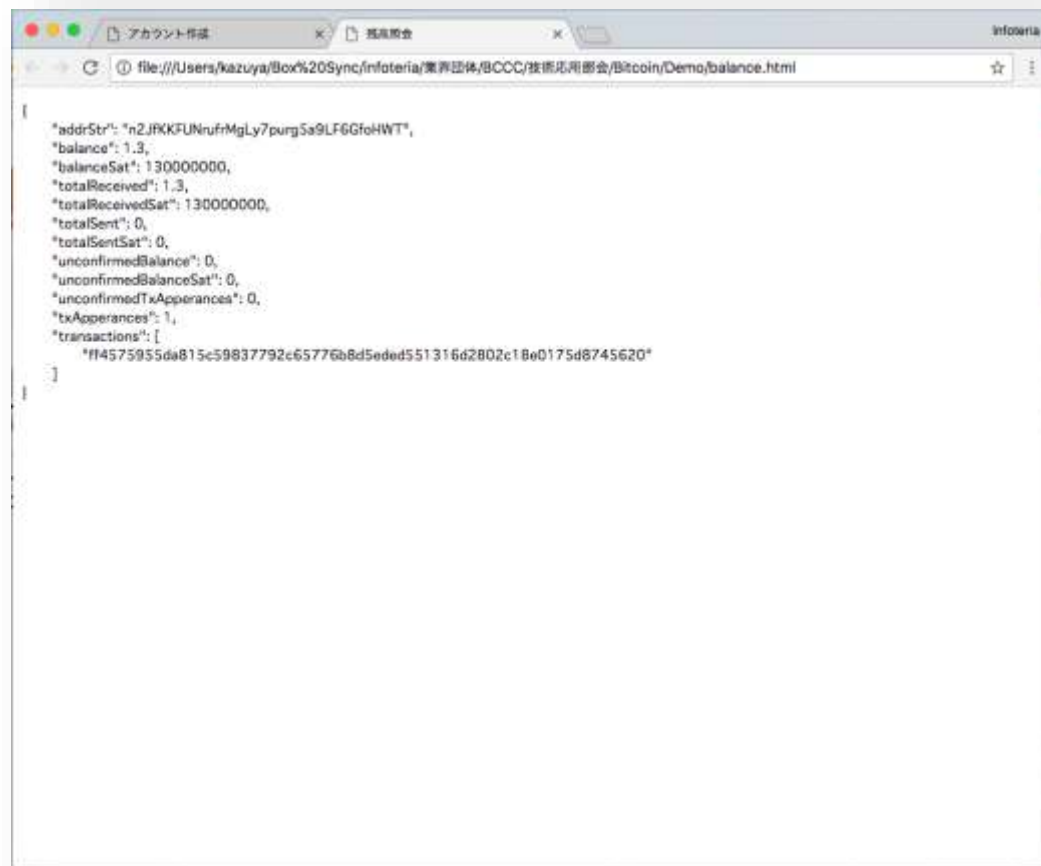
ご自身のBitcoinアドレスに置き換え
てください

いい感じでインデントしなければ
タブやスペースなどに置き換えてください

残高照会③ ～ 実行 ～



- 作成したHTMLとJSファイルを保存して、ブラウザでHTMLを実行してみよう。



```
{
  "addrStr": "n2JPKKFUNrfrMgLy7purg5a9LF6GfoHWT",
  "balance": 1.3,
  "balanceSat": 130000000,
  "totalReceived": 1.3,
  "totalReceivedSat": 130000000,
  "totalSent": 0,
  "totalSentSat": 0,
  "unconfirmedBalance": 0,
  "unconfirmedBalanceSat": 0,
  "unconfirmedTxAppearances": 0,
  "txAppearances": 1,
  "transactions": [
    "ff4575955da815c59837792c65776b8d5eded551316d2802c18e0175d8745620"
  ]
}
```

送金してみよう

送金するためには



① 相手のアドレスを知る

✦ あたりまえですが、送金相手のアドレスを知っていないと送金できません。

② UTXOを探す

✦ トランザクションにはマイニングで手に入れていない限り、かならず「入力」が必要です。

✦ まず「出力」されていない自分宛てのトランザクション（UTXO : Unspent Transaction Output）を探す必要があります。

③ トランザクションを作る

✦ トランザクションのプロトコルに合わせてデータを整形します。

④ ブロードキャストする

✦ ブロックチェーンのネットワークに送付して、合意形成をリクエストします。

送金してみよう① ～送金先の作成～

- 先程行った「アカウントをつくろう」で使ったHTMLを再読込して、アカウントをもうひとつ作りましょう。こちらを送金先とします。
- 他の人に送ってみたい方は、「アドレス」を教えてください。



送金してみよう② ～ HTML準備 ～



- 前と同じようにHTMLファイルを新規作成して、以下のような感じでJQuery と Bitcore を使えるようにしましょう。

sendcoin.html

```
<!DOCTYPE html>
<html lang="jp">
  <head>
    <meta charset="utf-8">
    <title>送金</title>
    <script type="text/javascript" src="https://code.jquery.com/jquery-3.2.1.min.js"></script>
    <script type="text/javascript" src="bitcore-lib.min.js"></script>
  </head>
  <body>
    <div id="contents"></div>
    <script type="text/javascript" src="sendcoin.js"></script>
  </body>
</html>
```

文字コードは環境に合わせて！

送金してみよう③ ～ JS準備 ～



- HTMLと同じフォルダにJSファイルを新規作成してスクリプトを書いていきましょう。

sendcoin.js

```
var endpoint = "https://testnet.blockexplorer.com";  
var fromAddress = "送金元のアドレス";  
var privateKey = "送金元の秘密鍵";  
var toAddress = "送金先のアドレス";  
var feeCoin = 10000;  
var sendCoin = 100000;
```

手数料です。単位は BTC ではなく、satoshi です。
適当な額を設定します。（変更しても良いです。）

送金額です。単位は BTC ではなく、satoshi です。
適当な額を設定します。（変更しても良いです。）

送金してみよう④ ～ UTXO取得 ～



- 残高照会と同じように、RESTでUTXOを取得します。

sendcoin.js (前のページの続き)

```
$.ajax({  
  type: "get",  
  url: endpoint + "/api/addr/" + fromAddress + "/utxo" ,  
  dataType: 'json'  
}).done(function(response){  
  createTransaction(response);  
}).fail(function(errordata){  
  console.log(data);  
});
```

取得成功したら、トランザクション作成のFunction
(次のページで説明) を呼びます。

送金してみよう⑤ ～ トランザクション作成 ～



- BitCoreライブラリを使って、トランザクションを作ります。

sendcoin.js (前のページの続き)

```
function createTransaction(utxo){  
  //ライブラリを読み込み  
  var bitcore = require('bitcore-lib');  
  //トランザクション作成  
  var transaction = new bitcore.Transaction()  
    .fee(feeCoin)  
    .from(utxo)  
    .to(toAddress, sendCoin)  
    .change(fromAddress)  
    .sign(privateKey);  
  //ブロードキャスト  
  broadcast(transaction);  
}
```

Fee : 手数料
From : 送信していないトランザクション (UTXO)
To : 宛先と送金額
Change : おつり
Sign : 署名

取得成功したら、ブロードキャストのFunction
(次のページで説明) を呼びます。

送金してみよう⑥ ～ ブロードキャスト ～



- RESTを使って署名付きトランザクションを送付します。

sendcoin.js (前のページの続き)

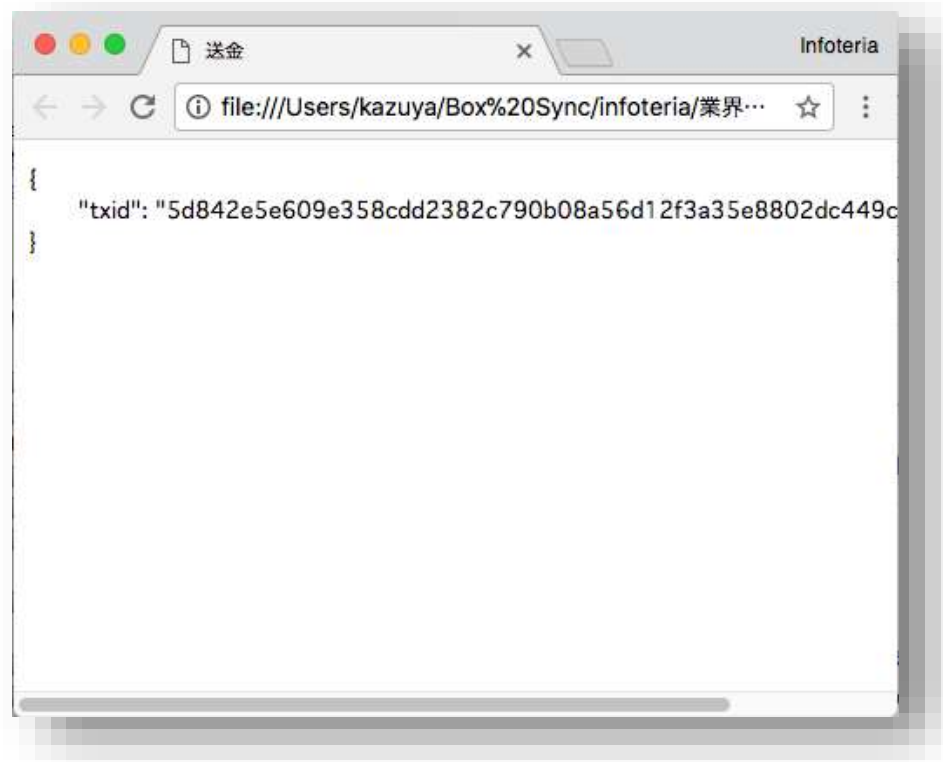
```
function broadcast(transaction){
  $.ajax({
    type: "post",
    url: endpoint + "/api/tx/send" ,
    contentType: 'application/json',
    dataType: 'json',
    data:JSON.stringify({rawtx:transaction.toString()})
  }).done(function(response){
    $("#contents").text(JSON.stringify(response,null,"¥t"));
  }).fail(function(errordata){
    console.log(errordata);
  });
}
```

成功したら結果を画面表示

送金してみよう⑦ ～実行～



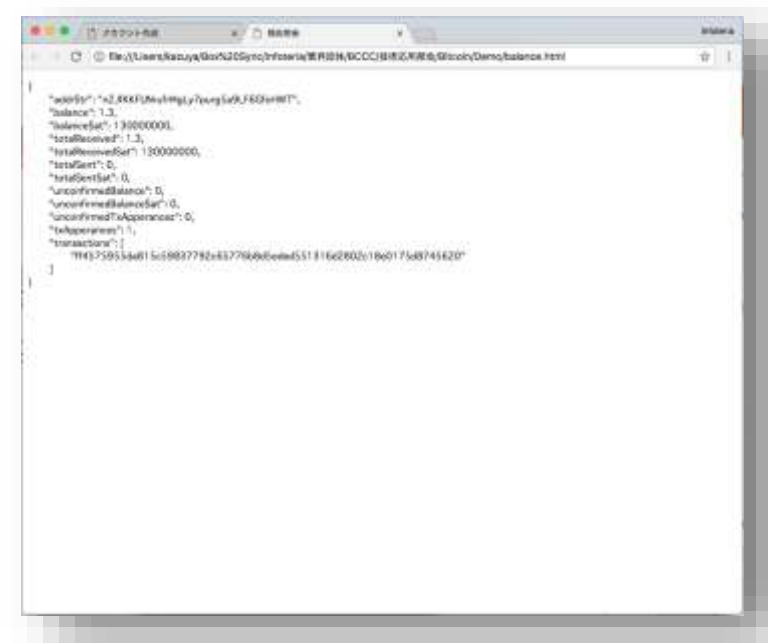
- 作成したHTMLとJSファイルを保存して、ブラウザでHTMLを実行してみよう。
- トランザクションのIDが返ってきます。



送金してみよう⑧ ～結果確認～



- Blockchain Explorer、もしくは先程作った残高確認のHTMLで結果を確認してみましょう。
- なお、送金直後はブロックに取り込まれていない状態の「Unconfirmed Balance」になります。



**ブロックチェーンに
ラクガキしてみよう**

レッツチャレンジ！



- 送金のトランザクションを少し変えるだけです！
- Transactionの宛先「.to(宛先アドレス,送金額)」の代わりにデータ追加「.addData(文字列)」を使います。
- 文字の上限があるので、あまり長いものは書き込めません。
- UTF8の16進数でトランザクションに書き込まれます。
(OP_RETURN について調べてみるのも良いでしょう)
- 結果はブロックチェーンエクスプローラー等でトランザクションを見てみましょう。