動作環境

Windows10-64bit

使用手順

1. BotCC.zip を任意の場所に解凍して下さい。
2. ApiKey,ApiSecret を取得し、同梱の ApiEncryptDecrypt.exe を起動し
   暗号化して下さい。(必要な権限は「参照」「取引」です。**「出金」は無効**にしておいて下さい)
3. 次ページ以降を参考に InitParam.json を編集してください。
4. ブラウザで CoinCheck の取引画面を表示させて下さい。(確認用)
5. BotCC.exe を起動して、設定した通りに動作しているかブラウザ画面および取引ログで確認して下さい。(タスクスケジューラー登録での使用を想定しているためタスクバーへは表示されません)
6. タスクスケジューラーへ登録してください。(都度手動で起動しても構いません)

※ 損切りは手動になります。ブラウザの取引画面で該当の注文をキャンセルして下さい。また、前注文においてチェック回数内に約定しない場合、注文はキャンセルされます。部分約定であっても未約定分の注文はキャンセルされます。金額・数量が大きい場合は部分約定が発生しやすいです。前注文の約定チェック回数も大きい値にして下さい。(フロー参照)

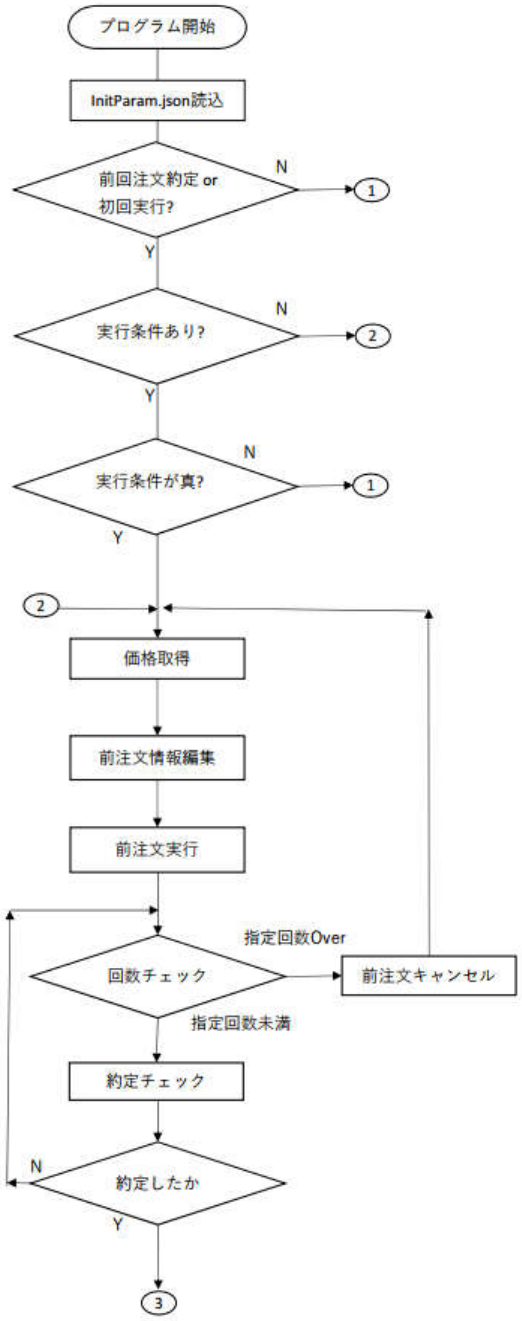
使用許諾

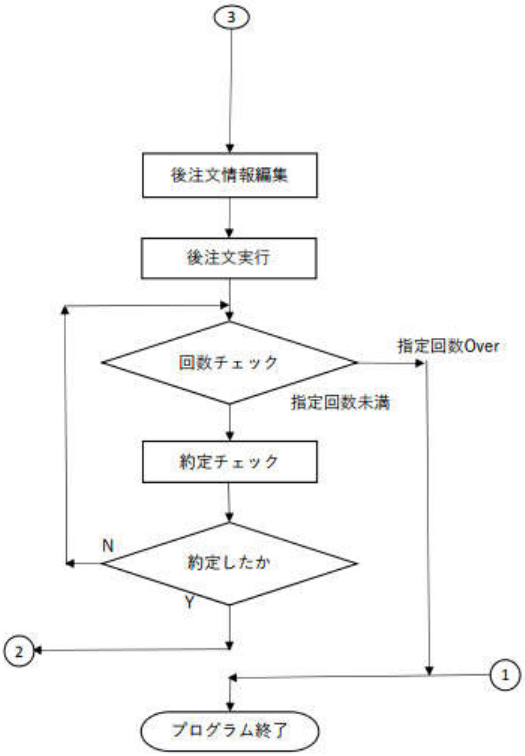
・BotCC.exe はフリー・ソフトウエアです。個人使用、業務使用に関わらず自由に使用してかまいません。

・BotCC.exe の動作に必要なファイルが含まれた形であれば、自由に複製し、頒布してかまいません。

・ソフトウエアは十分にテストをしていますが、お使いのパソコン環境や、プログラムの不具合などによって問題が生じる場合があります。 それ により損害が生 じても、損害に対する保証は出来かねますので、あらかじめご了承ください。

BotCC.exe の処理の流れ





InitParam.json の各項目について

項目名	説明
initValue	前注文時の基軸通貨の数量もしくは通貨の数量
tradePair	通貨ペア.通貨_(アンダーバー)基軸通貨 形式で設定します.
baseCurrOrAmount	initValue で設定した数量が基軸通貨の数量か通貨の数量か設定します.(“BaseCurr”もしくは”Amount”)
fixBaseCurrOrAmount	後注文でどちらの数量を固定するか設定します.(“BaseCurr”もしくは”Amount”)
maeSabun	前注文の注文価格と取得した価格 (売り注文は最低買価、買い注文は最高買価) の差分を設定します.
atoSabun	前注文の注文価格と後注文の注文価格の差分を設定します.
maeOrderType	前注文の注文種別.”BUY”か”SELL”を設定します.
maeOrderCloseChkCnt	前注文に対する約定チェックを行う回数を設定します.
maeOrderCloseChkInt	前注文の約定チェックを行う間隔(秒)を設定します
atoOrderCloseChkCnt	後注文に対する約定チェックを行う回数を設定します.
atoOrderCloseChkInt	後注文の約定チェックを行う間隔(秒)を設定します
runCntMax	PG の実行で(前注文～後注文を)何回繰り返すかを設定します。
orderNoFileName	直近注文の注文番号を記録するファイルをフルパスで設定します.
tradeLogFld	取引履歴を保存するフォルダをフルパスで設定します.
tanizaki	暗号化された API Key を設定します.
goma	暗号化された API Secret を設定します.
startPrice	PG 実行を開始する条件価格を設定します。次項の startOverUnder と併せて使用します。
startOverUnder	PG 実行開始の条件を設定します。前項の startPrice 以上(over)か、以下(under)かを指定します。実行開始条件を使用しない場合は”None”もしくは””(空白)を設定してください。

InitParam.json の設定例

価格が上昇すると予想した場合

(1)

```
  "initValue": 100000,
  "tradePair": "btc_jpy",
  "baseCurrOrAmount": "BaseCurr",
  "fixBaseCurrOrAmount": "Amount",
  "maeSabun": 0,
  "atoSabun": 1000,
  "maeOrderType": "buy",
```

・・・・・・・・・

（価格取得=1,000,000 JPY/BTC の場合）

・前注文の内容

注文種別:BUY(買い)

注文価格: 1000000

数　　量: 0.1(=100000/1000000)

収　　益:(1001000*0.1)-(1000000*0.1)=100JPY

0.1-0.1=0BTC

・後注文の内容

注文種別:SELL(売り)

注文価格:1001000(=1000000+1000)

数　　量:0.1

(2)

```
  "initValue": 100000,
  "tradePair": "btc_jpy",
  "baseCurrOrAmount": "BaseCurr",
  "fixBaseCurrOrAmount": " BaseCurr ",
  "maeSabun": -10000,
  "atoSabun": 10000,
  "maeOrderType": "buy",
```

・・・・・・・・・

（価格取得=1,000,000 JPY/BTC の場合）

・前注文の内容

注文種別:BUY(買い)

注文価格: 990000(=1000000-10000)

数　　量: 0.101(=100000/990000)

収　　益:100000-100000=0JPY

0.101-0.1=0.001BTC

・後注文の内容

注文種別:SELL(売り)

注文価格:1000000(=990000+10000)

数　　量:0.1(=100000/1000000)

価格が下落すると予想した場合

(3)

```
"initValue": 0.1,
"tradePair": "btc_jpy",
"baseCurrOrAmount": " Amount ",
"fixBaseCurrOrAmount": "Amount",
"maeSabun": 0,
"atoSabun": -1000,
"maeOrderType": "sell",
```

・・・・・・・・・

（価格取得=1,000,000 JPY/BTC の場合）

・前注文の内容

注文種別: SELL(売り)

注文価格: 1000000

数　　量: 0.1

・後注文の内容

注文種別: BUY(買い)

注文価格:999000(=1000000-1000)

数　　量:0.1

収　　益:(1001000*0.1)-(999000*0.1)=100JPY

0.1-0.1=0BTC

(4)

```
"initValue": 0.1,
"tradePair": "btc_jpy",
"baseCurrOrAmount": " Amount ",
"fixBaseCurrOrAmount": " BaseCurr ",
"maeSabun": 10000,
"atoSabun": -10000,
"maeOrderType": "sell",
```

・・・・・・・・・

（価格取得=1,000,000 JPY/BTC の場合）

・前注文の内容

注文種別: SELL(売り)

注文価格: 1010000(=1000000+10000)

数　　量: 0.1

注文金額: 101000(=1010000*0.1)

収　　益: 1010000-1010000=0JPY

0.101-0.1=0.01BTC

・後注文の内容

注文種別: BUY(買い)

注文価格:1000000

数　　量:0.101(=101000/1000000)

価格下落時の損失軽減を目的とする場合

(5)

"initValue": 0.1,
"tradePair": "btc_jpy",
"baseCurrOrAmount": " Amount ",
"fixBaseCurrOrAmount": "Amount",
"maeSabun": 0,
"atoSabun": -1000,
"maeOrderType": "sell",
・・・・・・・・・
"startPrice": 980000,
"startOverUnder": "under"

現在価格が1,000,000JPY → 980,000JPY → 950,000JPY

980,000JPY以下になったら下の処理を繰り返す。

・前注文の内容

注文種別: SELL(売り)

注文価格: 979000(*)

数　　量: 0.1

収　　益:(979000*0.1)-(978000*0.1)=100JPY

0.1-0.1=0BTC

・後注文の内容

注文種別: BUY(買い)

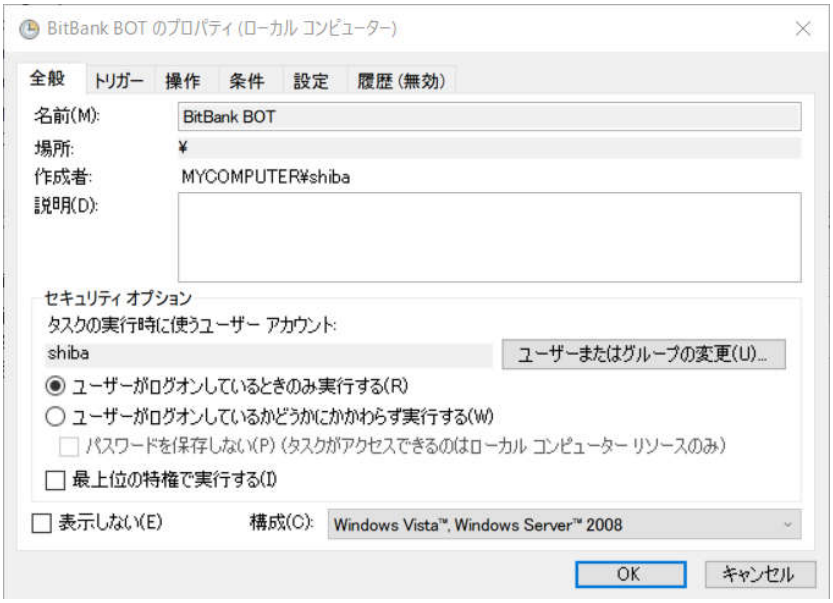
注文価格:978000(=979000-1000)

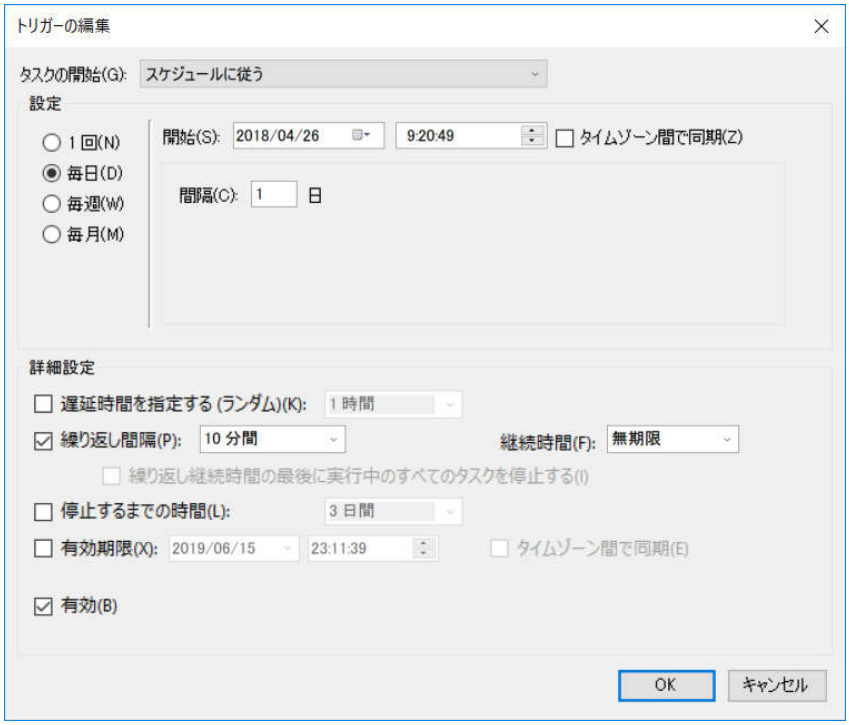
数　　量:0.1

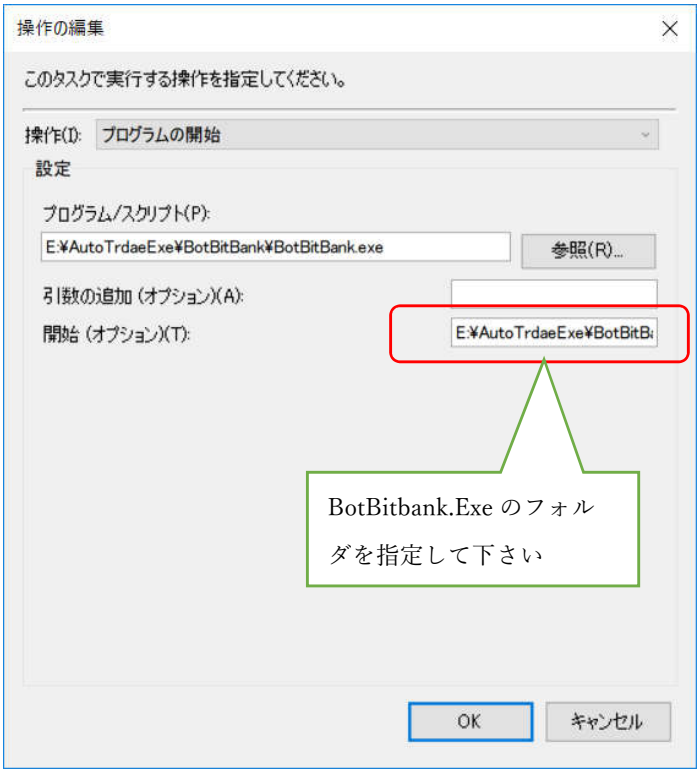
(*) 実際はその時点で取得した最高買値

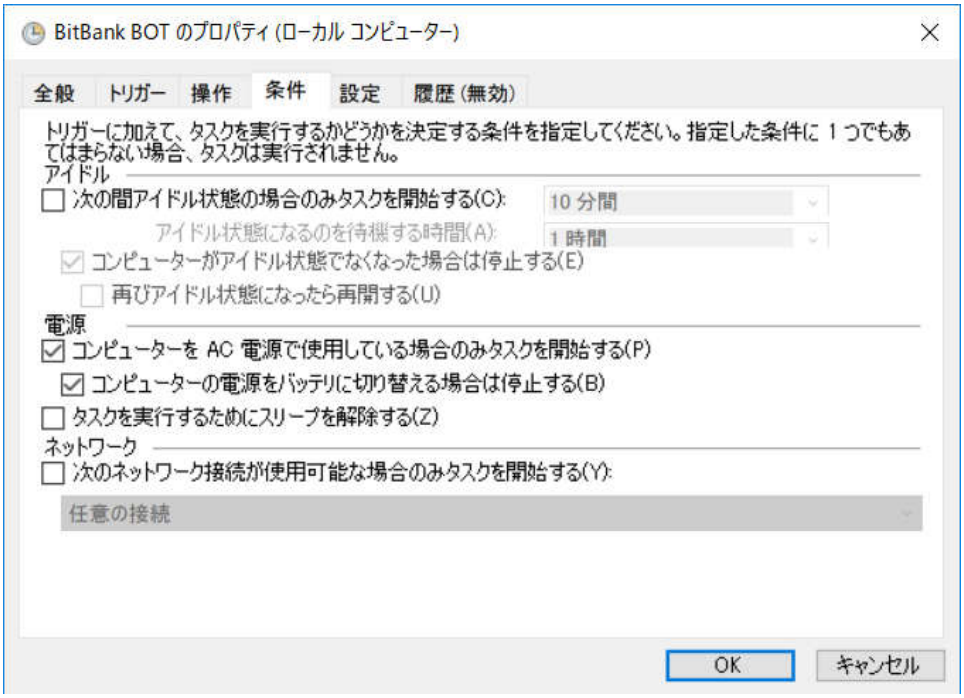
タスクスケジューラーの設定例

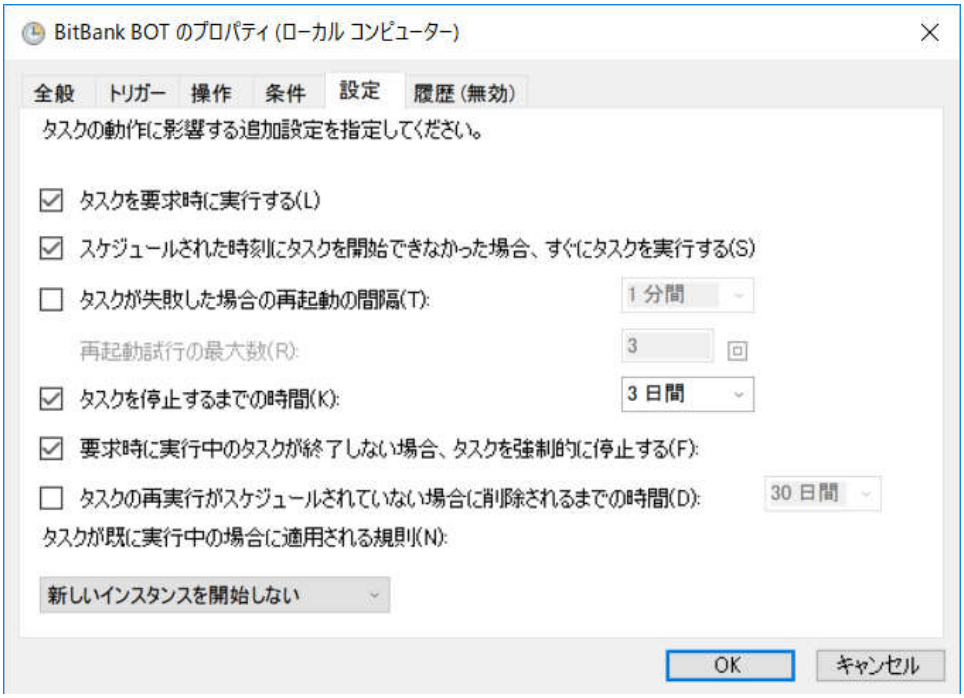
（各自の設定内容に合わせ調整して下さい。）






操作の編集
このタスクで実行する操作を指定してください。
操作(I): プログラムの開始
設定
プログラム/スクリプト(P):
E:¥AutoTrdaeExe¥BotBitBank¥BotBitBank.exe
参照(R)...
引数の追加 (オプション)(A):
開始 (オプション)(T):
E:¥AutoTrdaeExe¥BotBitB
BotBitbank.Exe のフォルダを指定して下さい
OK
キャンセル


BitBank BOT のプロパティ (ローカル コンピューター)
全般
トリガー
操作
条件
設定
履歴 (無効)
トリガーに加えて、タスクを実行するかどうかを決定する条件を指定してください。指定した条件に 1 つでもあてはまらない場合、タスクは実行されません。
アイドル
次の間アイドル状態の場合のみタスクを開始する(C):
10 分間
アイドル状態になるのを待機する時間(A):
1 時間
コンピューターがアイドル状態でなくなった場合は停止する(E)
再びアイドル状態になったら再開する(U)
電源
コンピューターを AC 電源で使用している場合のみタスクを開始する(P)
コンピューターの電源をバッテリに切り替える場合は停止する(B)
タスクを実行するためにスリープを解除する(Z)
ネットワーク
次のネットワーク接続が使用可能な場合のみタスクを開始する(Y):
任意の接続
OK
キャンセル


BitBank BOT のプロパティ (ローカル コンピューター)
全般
トリガー
操作
条件
設定
履歴 (無効)
タスクの動作に影響する追加設定を指定してください。
タスクを要求時に実行する(L)
スケジュールされた時刻にタスクを開始できなかった場合、すぐにタスクを実行する(S)
タスクが失敗した場合の再起動の間隔(T):
1 分間
再起動試行の最大数(R):
3
タスクを停止するまでの時間(K):
3 日間
要求時に実行中のタスクが終了しない場合、タスクを強制的に停止する(F)
タスクの再実行がスケジュールされていない場合に削除されるまでの時間(D):
30 日間
タスクが既に実行中の場合に適用される規則(N):
新しいインスタンスを開始しない
OK
キャンセル

不具合・取引所追加リクエスト・要望等　問い合わせ先

Mail　：　taroko.honpo@gmail.com
Twitter：https://twitter.com/THonpo
HP　　：https://github.com/TarokoHonpo/Namboja

寄付について

　儲かったら寄付していただけると、とてもうれしいです。

　（寄付はあくまで任意であり、あなたのご好意です。強制ではありません。）

通貨	アドレス
BTC	15MugbuE7rzJx7jr4XGW7YmdVqvZXhm1x5
MONA	MBfdDfNi8e472DpDyup7YkhJMFDWW1avRH
LTC	LWp7sHz95kibV6epw55QXqThZg3mBjg1fn
ETH	0xE7846cda44a15fD78B9F108F0B4d5bf38B492585
NANJ	0xB616E5606ccc9582c8dE155E6d95316969b888e0

　上記以外の通貨（塩漬け状態の草コインも可）でもご連絡いただければ用意いたします。