Code Mode Documentation - 1/3

baccarat, blackjack, blue_samourai, diamonds, cases, flip, general, engine_functions, custom_functions

---

system 1 June 10, 2025, 10:00am

General

All programming is done in JavaScript. You're allowed to use all the features, which JavaScript offer you, in this editor. Variable and function names are case-sensitive!

You can find example codes for all games in the sections below. Please note, that these are just examples to give you some inspiration and are not guaranteed to make profit!

Variables

The following variables are available in all games:

- `game`
  Type: string
  Required: true
  Possible values: baccarat, blackjack, slots-samurai, cases, coinflip, diamonds, dice, tower, hilo, keno, limbo, mines, plinko, pump, rock-paper-scissors, roulette, slots-scarab, snakes, slots-tomeoflife,video-poker, wheel

- `casino`
  Type: string (read-only)
  Possible values: BCHGAMES, STAKE, STAKE_US, PRIMEDICE, SHUFFLE, WOLFBET, GOATED, CHIPSGG

- `minBetSize`
  Type: float (read-only)
  The minimum bet size for the currently active currency.

- `username`
  Type: string (read-only)
  Username of the currently logged in casino account.

- `lastBet`
  Type: object (read-only)

- `asyncMode`
  Type: boolean (default: false)

Use async mode on your own risk! If turned on, bets will be made asynchronously, which means that when switching to normal mode, it can take some more bets until it completely finished all bets made in async mode. Also we do not recommend using async mode, when increasing bet size based on losses or wins. Only use this when you know what you're doing!

- `lookupBetIds`
  Type: boolean (default: false)
  Set this to true if you want the bot to lookup the bet IDs (casino:...). We recommend to only set this to true if it's really necessary for your strategy.

- `isZeroBettingAllowed`
  Type: boolean (read-only)
  Indicates if betting with 0 bet size is supported in the casino you're playing in.

- `isSimulationMode
  Type: boolean (read-only)
  Check if the current mode is on simulation mode or not.

- `currentStreak`
  Type: integer (read-only)
  Positive on a win streak, negative on a loss streak.

- `rollNumber`
  Type: integer (read-only)

- `balance`
  Type: float (read-only)

- `currency`
  Type: string (read-only)

- `getConversionRate()`
  Type: object (read-only)
  Conversion rates for all supported crypto and FIAT currencies . When called without parameters, it retrieves the conversion rate based on the currently active cryptocurrency and fiat currency (defaulting to USD). The function also accepts two optional parameters: the cryptocurrency code and the fiat currency code. Example: (getConversionRate()`will get the currency rate for the currently active crypto in USD` (getConversionRate(currency,"EUR")`will get the currency rate for the currently active crypto in EUR`
  (getConversionRate("TRX","EUR")` will get the currency rate for the TRX crypto in EUR

- `profit`
  Type: float (read-only)

- `highestProfit`
  Type: float (read-only)
- `lowestProfit`
  Type: float (read-only)
- `wagered`
  Type: float (read-only)
- `vaulted`
  Type: float (read-only)
- `wins`
  Type: integer (read-only)
- `losses`
  Type: integer (read-only)
- `rtp`
  Type: float (read-only)
- `leftSeedResets`
  Type: float (read-only)
- `bestB2bMultipliers`
  Type: array (read-only)
  Array of the best B2B multipliers, sorted by highest multiplier.
- `betsPerSecond`
  Type: integer (read-only)
- `highestBet`
  Type: float (read-only)
- `highestLoss`
  Type: float (read-only)
- `highestMultipliers`
  Type: array (read-only)
  Array of the highest won multipliers, sorted by highest multiplier.
- `maxDrawdown`
  Type: float (read-only)
  Maximum draw down, meaning the highest distance between the highest profit and the lowest profit.

- `profitPerHour`
  Type: float (read-only)
  Profit projection based on betting timer and current profit.

- `bestWinStreaks`
  Type: array (read-only)
  Array of the best win streaks, sorted by highest win streak.

- `worstLossStreaks`
  Type: array (read-only)
  Array of the best loss streaks, sorted by highest loss streak.

- `timeRunning`
  Type: integer (read-only)
  Amount of seconds betting is running, analogue to the timer that's shown above the bet list

Engine functions

- `engine.start()`
  Starts betting, equal to pressing the Start button.
- `engine.stop()`
  Stops betting, equal to pressing the Stop button.
- `engine.pause()`
  Pauses betting, equal to pressing the Pause button.
- `engine.resume()`
  Resumes betting, equal to pressing the Resume button.
- `engine.onBetPlaced(callback)`
  Sets the callback function (supports async), which is executed after a bet was placed. The first parameter in the callback function is the lastBet object. Find more information about the lastBet object of every single game in the tabs below.
- `engine.onGameRound(callback)`
  Sets the callback function (supports async), which is executed for sub-rounds of a game (e.g. in BlackJack or Hilo). Find more information about the lastBet object of every single game in the tabs below.
- `engine.onBettingStopped(callback)`
  Sets the callback function, which is executed, when betting was stopped. The first parameter in the callback function is a boolean, which is true when the betting was stopped manually (e.g. when pressing the Stop button or calling engine.stop()).

Custom functions

- `log(var1 [, var2, ..., varN])`
  Example: `log(`Bet size: ${betSize}`)`
  Example: `log(lastBet)`
  Example: `log(balance, profit)`
  Print information in the console tab.

  If the first argument is a HEX color code, this will be the color of the log message.
  Example: `log('#eb4034', `Bet size: ${betSize}`)`

- `clearConsole()`
  Clear messages in the console tab.

- `notification(message [, type = 'info'])`
  Example: `notification('Profit target reached!', 'success')`
  Possible values for the type variables are: error, success, warning, info (default)

- `resetStats()`
  Resets all statistics. Same behaviour, like when you click on the reset button in the statistics window.

- `resetSeed()`
  Example: `resetSeed()`
  Example: `resetSeed('myCustomClientSeed')`
  Creates a new server/client seed pair.

- `logToFile(fileName, content)`
  Example: `logToFile('bets.csv', `${lastBet.nonce};${lastBet.payoutMultiplier}`)`
  Appends the passed in content to the given file name. If the file doesn't exist, it will be created. Files are stored in `~/Documents/qBot/userLogs`.

- `chanceToMultiplier(chance)`
  Example: `chanceToMultiplier(49.5); // Returns 2`
  Converts chance from dice to multiplier.

- `depositToVault(amount)`
  Example: `depositToVault(0.001);`
  Deposits the given amount to the vault. Currently selected currency is used.

- `playHitSound()`
  Example: `playHitSound();`
  Plays the currently configured hit sound.

- `setSimulationBalance(amount)`
  Example: `setSimulationBalance(100);`
  Sets the simulation balance (Q$ FUN).

- `setSimulationNonce(nonce)`
  Example: `setSimulationNonce(1);`
  Sets the simulation nonce.

- `shuffle(array)`
  Example: `shuffledArray = shuffle([0, 1, 2, 3, 4]); // [3, 4, 1, 0, 2]`
  Randomizes the order of the input array.

- `getVipProgress()`
  Example: `await getVipProgress()`
  Returns the current VIP progress in the following format:

```
{
    flag: 'None',
    progress: 50.00,
    nextFlag: 'Bronze',
    wagerLeft: 5000
}
```

Baccarat

Variables

- `bankerBetSize`
  Type: float
  Required: true
- `playerBetSize`
  Type: float
  Required: true
- `tieBetSize`
  Type: float
  Required: true

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 2,
  "nonce": 1027,
  "active": false,
  "game": "baccarat",
  "win": true,
  "amount": 1e-7,
  "fiatAmount": "$0.00",
```

```
  "payoutMultiplier": 2,
  "b2bMultiplier": 2,
  "payout": 2e-7,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "playerCards": [
      {"suit": "D", "rank": "Q"},
      {"suit": "C", "rank": "5"},
      {"suit": "C", "rank": "9"}
    ],
    "bankerCards": [
      {"suit": "S", "rank": "J"},
      {"suit": "D", "rank": "3"},
      {"suit": "S", "rank": "8"}
```


Code example

```
// Fibonnaci sequence betting

// resetStats();
// resetSeed();

game = 'baccarat';
bankerBetSize = 0.00000000;
playerBetSize = 0.00000010;
tieBetSize = 0.00000000;
initialPlayerBetSize = playerBetSize;
fibonacciSequence = [
    1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377,
    610, 987, 1597, 2584, 4181, 6765, 10946, 17711,
    28657, 46368, 75025, 121393, 196418, 317811
];

engine.onBetPlaced(async (lastBet) => {
    if (lastBet.win) {
        playerBetSize = initialPlayerBetSize;
    } else {
        playerBetSize = initialPlayerBetSize * fibonacciSequence[Math.abs(curren
    }
});

engine.onBettingStopped((isManualStop, lastError) => {
    playHitSound();
    log(`Betting stopped!`);
```

Blackjack

Variables

- `betSize`
  Type: float
  Required: true
- `sideBetPerfectPairs`
  Type: float
  Required: true
  Note: Side bets are optional and not supported by every casino.
- `sideBet213`
  Type: float
  Required: true
  Note: Side bets are optional and not supported by every casino.

Functions

- `blackJackPerfectNextAction(currentBet, playerHandIndex, strategy = 'easy', double = 'any')`
  Example: `blackJackPerfectNextAction(currentBet, playerHandIndex, 'easy', 'any');`
  Returns the next action (see onGameRound return constants below), depending on the given parameters.

  The following string parameters can be passed as strategy (3rd parameter):

  - `easy`
  - `simple`
  - `advanced`
  - `exactComposition`
  - `bjc-supereasy`
  - `bjc-simple`
  - `bjc-great`

  The following string parameters can be passed as doubling option (4th parameter):

  - `none`
  - `10or11`
  - `9or10or11`
  - `any`

  Please find further information about how the strategies work here: **github.com/gsdriver/blackjack-strategy**

- `blackJackBetMatrixNextAction(currentBet, playerHandIndex, betMatrix)`
  Returns the next action (see onGameRound return constants below), depending on the given bet matrix.

  The following commands are possible to set in the bet matrix values:

  - `H` = Hit
  - `S` = Stand
  - `P` = Split
  - `D` = Double or Hit
  - `DS` = Double or Stand

  The following code example shows the bet matrix for the perfect strategy:

```
betMatrix = {
    "hard": {
        "4": {"2": "H", "3": "H", "4": "H", "5": "H", "6": "H", "7": "H", "8"
        "5": {"2": "H", "3": "H", "4": "H", "5": "H", "6": "H", "7": "H", "8"
        "6": {"2": "H", "3": "H","4": "H", "5": "H", "6": "H", "7": "H", "8":
        "7": {"2": "H", "3": "H", "4": "H", "5": "H", "6": "H", "7": "H", "8"
        "8": {"2": "H", "3": "H", "4": "H", "5": "H", "6": "H", "7": "H", "8"
        "9": {"2": "H", "3": "D", "4": "D", "5": "D", "6": "D", "7": "H", "8"
        "10": {"2": "D", "3": "D", "4": "D", "5": "D", "6": "D", "7": "D", "8
        "11": {"2": "D", "3": "D", "4": "D", "5": "D", "6": "D", "7": "D", "8
        "12": {"2": "H", "3": "H", "4": "S", "5": "S", "6": "S", "7": "H", "8
        "13": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "H", "8
        "14": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "H", "8
        "15": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "H", "8
        "16": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "H", "8
        "17": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "S", "8
        "18": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "S", "8
        "19": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "S", "8
        "20": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "S", "8
        "21": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "S", "8
    },
    "soft": {
        "12": {"2": "H", "3": "H", "4": "H", "5": "D", "6": "D", "7": "H", "8
        "13": {"2": "H", "3": "H", "4": "H", "5": "H", "6": "D", "7": "H", "8
        "14": {"2": "H", "3": "H", "4": "H", "5": "D", "6": "D", "7": "H", "8
        "15": {"2": "H", "3": "H", "4": "H", "5": "D", "6": "D", "7": "H", "8
        "16": {"2": "H", "3": "H", "4": "D", "5": "D", "6": "D", "7": "H", "8
```

onGameRound return constants

- `BLACKJACK_DOUBLE`
  Type: string

- `BLACKJACK_HIT`
  Type: string

- `BLACKJACK_SPLIT`
  Type: string

- `BLACKJACK_STAND`
  Type: string

lastBet example

```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 2,
  "nonce": 1027,
  "active": true,
  "game": "blackjack",
  "win": false,
  "amount": 1e-7,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 0,
  "b2bMultiplier": 0,
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "player": [
      {
        "value": 17,
        "actions": ["deal"],
        "cards": [
          {"rank": "Q", "suit": "D"},
          {"rank": "7", "suit": "D"}
        ]
      }
    ],
```

Code example for built-in strategies

```
// Just plays Black Jack with the optimal strategy, doing martingale

// resetStats();
// resetSeed();

game = 'blackjack';
betSize = 0.00000001; // Bet size is always specified in crypto value, not USD!
sideBetPerfectPairs = 0.00000001; // Side bets are optional and are not supporte
```

```
sideBet213 = 0.00000001; // Side bets are optional and are not supported by all
initialBetSize = betSize;

engine.onBetPlaced(async (lastBet) => {
    if (lastBet.win) {
        if (lastBet.payoutMultiplier > 1) {
            // Only reset bet size, if we hit anything greater than 1x
            betSize = initialBetSize;
        }
    } else {
        betSize *= 2;
    }
});

engine.onGameRound((currentBet, playerHandIndex) => {
    const nextAction = blackJackPerfectNextAction(currentBet, playerHandIndex,

    if (nextAction === BLACKJACK_DOUBLE) {
        betSize *= 2;
```

Code example for bet matrix

```
// Just plays Black Jack with the perfect strategy, doing martingale

// resetStats();
// resetSeed();

game = 'blackjack';
betSize = 0.00000001; // Bet size is always specified in crypto value, not USD!
sideBetPerfectPairs = 0.00000001; // Side bets are optional and are not supporte
sideBet213 = 0.00000001; // Side bets are optional and are not supported by all
initialBetSize = betSize;

betMatrix = {
    "hard": {
        "4": {"2": "H", "3": "H", "4": "H", "5": "H", "6": "H", "7": "H", "8": '
        "5": {"2": "H", "3": "H", "4": "H", "5": "H", "6": "H", "7": "H", "8": '
        "6": {"2": "H", "3": "H","4": "H", "5": "H", "6": "H", "7": "H", "8": "H
        "7": {"2": "H", "3": "H", "4": "H", "5": "H", "6": "H", "7": "H", "8": '
        "8": {"2": "H", "3": "H", "4": "H", "5": "H", "6": "H", "7": "H", "8": '
        "9": {"2": "H", "3": "D", "4": "D", "5": "D", "6": "D", "7": "H", "8": '
        "10": {"2": "D", "3": "D", "4": "D", "5": "D", "6": "D", "7": "D", "8":
        "11": {"2": "D", "3": "D", "4": "D", "5": "D", "6": "D", "7": "D", "8":
        "12": {"2": "H", "3": "H", "4": "S", "5": "S", "6": "S", "7": "H", "8":
        "13": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "H", "8":
        "14": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "H", "8":
        "15": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "H", "8":
        "16": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "H", "8":
        "17": {"2": "S", "3": "S", "4": "S", "5": "S", "6": "S", "7": "S", "8":
```

Blue Samurai

Variables

- `betSize`
  Type: float
  Required: true

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 3,
  "nonce": 1027,
  "active": false,
  "game": "slots-samurai",
  "win": true,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 1.5,
  "b2bMultiplier": 1.5,
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "nextSpinType": "complete",
    "bonusSpinsRemaining": 0,
    "rounds": [
      {
        "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
        "type": "regular",
        "data": {
          "spinMultiplier": 0,
          "roundMultiplier": 0,
          "view": [
```


Code example

```
// Basic martingale example with stop on loss streak of 10

// resetStats();
// resetSeed();

game = 'slots-samurai';
betSize = 0.00000000;
```

```
initialBetSize = betSize;

engine.onBetPlaced(async (lastBet) => {
    if (currentStreak === -10) {
        log('Loss streak of 10 reached. Stopping...');
        engine.stop();
    }

    if (lastBet.win) {
        betSize = initialBetSize;
    } else {
        betSize *= 2;
    }
});

engine.onBettingStopped((isManualStop, lastError) => {
    playHitSound();
    log(`Betting stopped!`);
});
```

Cases

Variables

- `betSize`
  Type: float
  Required: true
- `difficulty`
  Type: string
  Required: true
  Possible values: easy, medium, hard, expert

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 6,
  "nonce": 1027,
  "active": false,
  "game": "cases",
  "win": true,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 1.6333333333333333,
  "b2bMultiplier": 1.6333333333333333,
  "payout": 0,
  "fiatPayout": "$0.00",
```

```
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "difficulty": "expert",
    "result": 2.00
  }
  "clientSeed": "yourClientSeed",
  "serverSeed": "XXX", // Simulation Mode only
  "serverSeedHashed": "XXX" // Live Mode only
}
```

Code example

```
// Randomly change difficulty after every bet

// resetStats();
// resetSeed();

game = 'cases';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!
initialBetSize = betSize;
difficulties = ['easy', 'medium', 'hard', 'expert'];
difficultyIndex = 0;
difficulty = difficulties[difficultyIndex];

engine.onBetPlaced(async (lastBet) => {
    difficulty = difficulties[++difficultyIndex];

    if (difficultyIndex + 1 === difficulties.length) {
        difficultyIndex = 0;
    }
});

engine.onBettingStopped((isManualStop, lastError) => {
    playHitSound();
    log(`Betting stopped!`);
});
```

Flip

Variables

- `betSize`
  Type: float
  Required: true

- `pattern`
  Type: array
  Required: true
  This must be an array, which holds ‘h’ or ‘t’ values. The length of the array implicitly defines the amount of rounds you want to play. The maximum amount of items is 20.

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 4,
  "nonce": 1027,
  "active": false,
  "game": "coinflip",
  "win": false,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 0,
  "b2bMultiplier": 0,
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  {
    "currentRound": 5,
    "payoutMultiplier": 31.36,
    "playedRounds": [
      "h",
      "h",
      "t",
      "h",
      "t"
    ],
    "flips": [
```


Code example

```
// Randomly change pattern after every win

// resetStats();
// resetSeed();

function randomPattern(rounds) {
    const pattern = [];

    for (let i = 0; i < rounds; i++) {
```

```
        // Generate random number between 0 and 1
        // If less than 0.5, it's heads ('h'), otherwise tails ('t')
        const flip = Math.random() < 0.5 ? 'h' : 't';
        pattern.push(flip);
    }

    return pattern;
}

game = 'coinflip';
betSize = 0.00000000; // Bet size is always specified in crypto value, not USD!

const rounds = 5;
pattern = randomPattern(rounds);

engine.onBetPlaced(async (lastBet) => {
    if (lastBet.win) {
        pattern = randomPattern(rounds);
```

Diamonds

Variables

- `betSize`
  Type: float
  Required: true

lastBet example


```
{
  "id": "XXXX-XXXX-XXXX-XXXX-XXXX",
  "iid": "casino:888888888",
  "rollNumber": 4,
  "nonce": 1027,
  "active": false,
  "game": "diamonds",
  "win": false,
  "amount": 0,
  "fiatAmount": "$0.00",
  "payoutMultiplier": 0,
  "b2bMultiplier": 2,
  "payout": 0,
  "fiatPayout": "$0.00",
  "currency": "eth",
  "dateTime": "2022-05-01T00:00:00.000Z",
  "deepLink": "https://stake.com/?betId=XXXX-XXXX-XXXX-XXXX-XXXX&modal=bet",
  "state": {
    "hand": [
      "blue",
      "orange",
```

```
      "yellow",
      "red",
      "green"
    ]
  },
  "clientSeed": "yourClientSeed",
  "serverSeed": "XXX", // Simulation Mode only
```

Code example

```
// Hunt B2B streak of 3 wins

// resetStats();
// resetSeed();

game = 'diamonds';
betSize = 0.00000000;
initialBetSize = betSize;

engine.onBetPlaced(async (lastBet) => {
    log(`Diamonds: ${lastBet.state.hand.join(', ')}`);

    if (lastBet.win) {
        betSize = lastBet.payout;
    } else {
        betSize = initialBetSize;
    }

    if (currentStreak === 3) {
        betSize = initialBetSize;
    }
});

engine.onBettingStopped((isManualStop, lastError) => {
    playHitSound();
    log(`Betting stopped!`);
});
```

Hi my code is not working